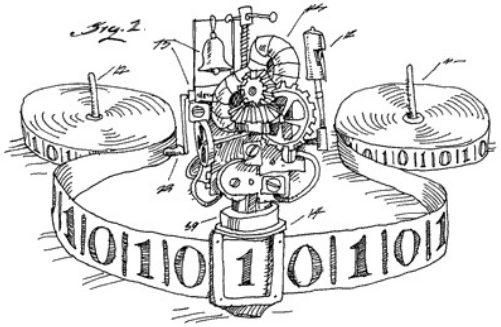
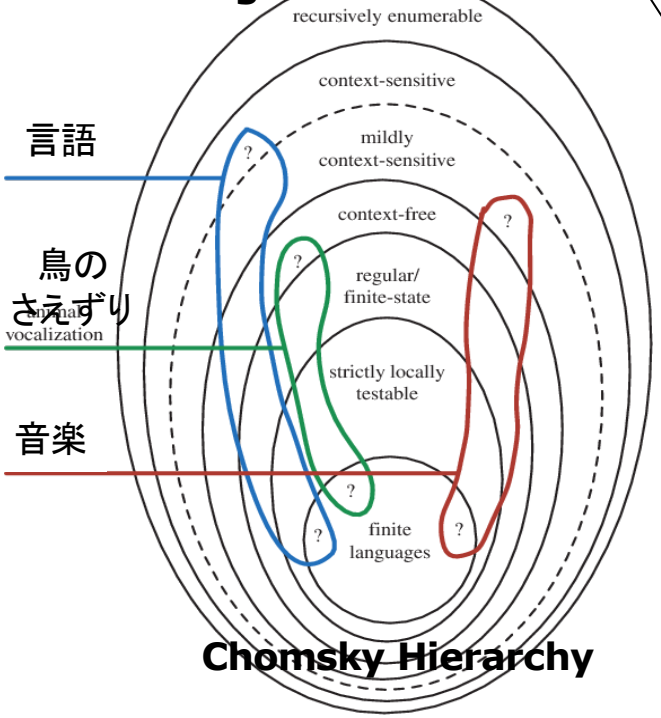
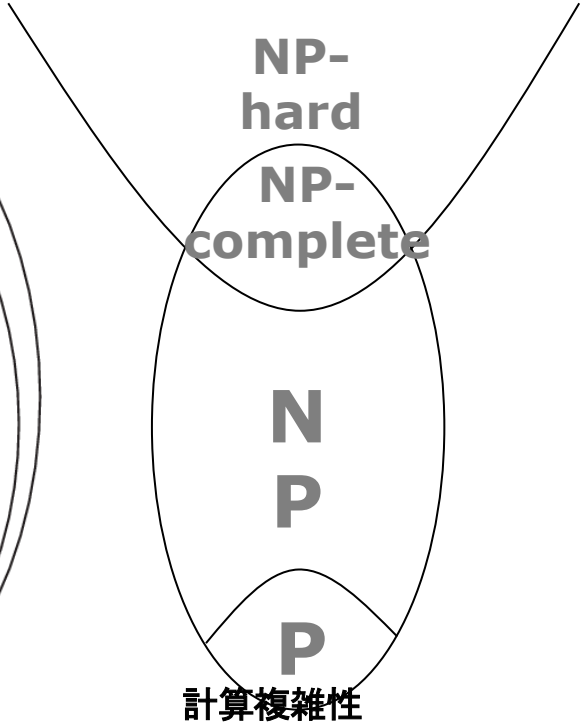
人工知能と複雑性理論



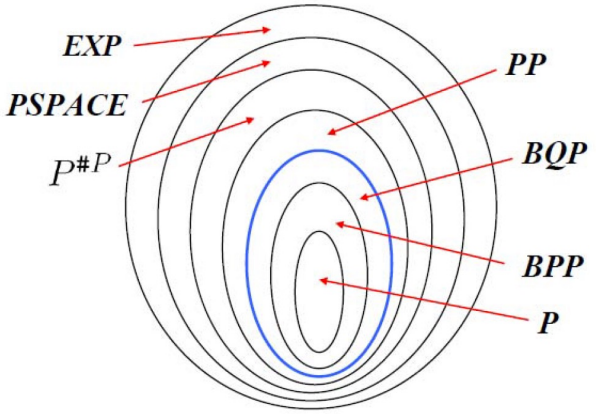
Turing Machine



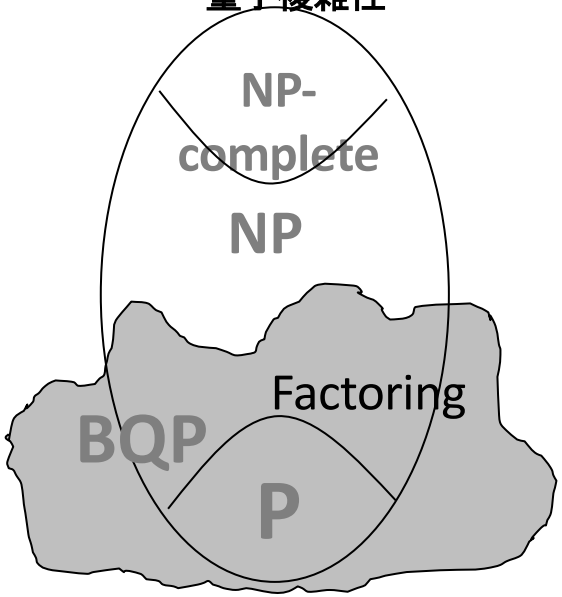
Chomsky Hierarchy



計算複雑性



量子複雑性



I propose to consider the question, "Can machines think?"
....
The original question, "Can machines think?" I believe to be too meaningless to deserve discussion. Nevertheless I believe that at the end of the century the use of words and general educated opinion will have altered so much that one will be able to speak of machines thinking without expecting to be contradicted.

-- Alan Turing

What's the main idea of this paper? As I read it, it's a plea against meat chauvinism. Sure, Turing makes some scientific arguments, some mathematical arguments, some epistemological arguments. But beneath everything else is a moral argument.

One can indeed give weighty and compelling arguments against the possibility of thinking machines. The only problem with these arguments is that they're also arguments against the possibility of thinking brains!

-- *Scott Aaronson*

Voice Two: "So who made the machines? That's who we want to contact."

Voice One: "They made the machines. That's what I'm trying to tell you. Meat made the machines."

Voice Two: "That's ridiculous. How can meat make a machine? You're asking me to believe in sentient meat."

Voice One: "I'm not asking you, I'm telling you. These creatures are the only sentient race in that sector and they're made out of meat.”

Terry Bisson “ They‘re Made out of Meat!"

はじめに

- 小論は、人口知能研究の今後の発展の方向を、数学的・物理学的認識可能性の理論でもある「複雑性理論」の成立・発展を中軸にして考察したものである。
- なぜ、数学的・物理学的認識が問題になるのか？
  もちろん、人工知能研究の基本的な問題は、「機械は考えることができるか？」という、チューリング以来の問題である。
- 「機械が考えること」の中に、感覚＝運動的な認識能力だけで無く、言語的な認識や数学的認識を含めると問題は急に難しくなる。ただ、こうした問題を避けるわけにはいかないと、筆者は考えている。また、人工知能は、理論的な構成物としてではなく、物理的なものとして実装されねばならない。
- 人間と機械の数学的・物理的認識の限界を（筆者は、両者は同じ限界を持つと考えている）、知ることは、すなわち、人工知能の認識の限界を知ることに他ならない。

はじめに

- 人工知能研究の現状からいうと、問題をかえって難しくしているだけと感じる人もいるかもしれない。ただ、「複雑性理論」を視野に入れると、未来についての見通しは、むしろ良くなると筆者は感じている。
- 「複雑性理論」の第一世代は、1950年代に定式化される「計算可能性理論」である。それは、数学的・形式的な抽象的なものでしかなかった。
- 「複雑性理論」の第二世代は、1970年代に始まる「計算複雑性理論」である。それは、コンピュータの爆発的普及に刺激を受け、コンピュータの能力を意識したものであったが、数学的・形式的なものであった。（もっとも、その基本問題である P=NP? 問題は、そうした性質を持ち続けている）

はじめに

- 「複雑性理論」の第三世代は、1980年代の「理論的可能性」としての「量子コンピュータ」の登場とともに始まる。
- この第三世代は、「複雑性理論」の発展としては、「量子複雑性理論」の成立として特徴付けられるのだが、その過程は、「情報過程＝物理過程」という認識によって支えられていた。
- 同時に、それは、「物理過程＝情報過程」という認識の成熟でもあった。ブラックホールやエンタングルメントのエントロピーの発見、ブラックホールでの「情報消失」問題からホログラフィック原理の提唱、AdS/CFT 対応の発見等々の現代物理学の発展の中心に「量子情報理論」と「複雑性理論」がある。
- こうした知見は、我々人間の（そして恐らくは機械の、それゆえ「人工知能」の）認識の限界が、どのようなものであるかを知らせている。

はじめに

- 「複雑性理論」は、我々人間が、この宇宙の中で知りうることについての「限界」の具体的で精緻な「境界付け」を与えてくれるのである。
- 我々の認識の「限界」が、こうして、ますます身近で具体的なものとして与えられていく反面、それでも、大きな謎は残っている。我々を拘束する、こうした「有限性」や「多項式時間」という「制限」の中で、我々は、なぜ自由に思考し、無限について数学的に考えることが出来るのであろうか？「謎」があることと、科学は矛盾しはしないのだ。
- スライドは、少し長く、一部は難しい表現も含まれているかもしれない。ただ、この問題領域での興味ふかいエピソードを沢山盛り込んだ。わからないところは飛ばして、面白そうなところだけ読んでもらっても、この世界の広がりは伝わると考えている。

Agenda

- はじめに
- **Part I**
  計算主義とコネクショニズム
- **Part II**
  計算可能性理論
- **Part III**
  計算複雑性理論
- **Part IV**
  量子複雑性理論
- **Part V**
  数学の「証明」をめぐって
- まとめにかえて

Agenda Part I
計算主義とコネクショニズム

- 計算主義とは何か？
- 「知能」の階層性
- ディープラーニングの「形式性」の認識能力と「生成能力」
  - LSTMの能力について　Hochreiter
  - RNNによる文の生成　Ilya Sutskever
  - RNNの驚くべき能力について　Karpathy

Agenda Part II
計算可能性理論

- 計算可能性理論
  - ゲーデルの不完全性定理
  - Church-Turing Thesis
- 計算主義の源流　TuringとChomsky
  - チューリングの問題提起
    「機械は考えることができるか？」
  - 文法の階層性　Chomsky Hierarchyについて
- 「有限の立場」と「実効的計算可能性」
  - 一般帰納関数の計算可能性　アッカーマン関数
  - Cantorの順序数
    - ゲンツェンの数論の無矛盾性証明

Agenda Part III
計算複雑性理論

- 計算複雑性理論
  - Gödel とNash
  - 計算複雑性理論とは何か
  - 計算複雑性理論の基本的クラス
- 複雑さについての多様なアプローチ
  - コルモゴロフの複雑性
  - チャイティンの不完全性定理
  - 我々に手の届かない有限について
Busy Beaver 問題
  - チャイティンとゲーデル

Agenda Part IV
量子複雑性理論

- 量子複雑性理論
  - Feynman
  - Deutsch
  - BQP Bernstein と Vazirani
  - Quantum Complexity Theory
  - Shor
  - Extended Church-Turing Thesis
  - NP-complete Problems and Physical Reality
- 物理学と量子情報理論

Agenda Part V

数学の「証明」をめぐって

- 数学の「証明」の問題
- 数学でのコンピューターの利用
- Voevodsky
- Univalent FoundationsとUniMath
- 数学とNP問題
- 複雑性理論でのクラス MA: Merlin-Arthur

Part I

計算主義とコネクショニズム

Agenda Part I
計算主義とコネクショニズム

- 計算主義とは何か？
- 「知能」の階層性
- ディープラーニングの「形式性」の認識能力と「生成能力」
  - LSTMの能力について　Hochreiter
  - RNNによる文の生成　Ilya Sutskever
  - RNNの驚くべき能力について　Karpathy

計算主義とは何か？

計算主義とは何か？

人工知能論における「計算主義」とは、人間の「知能」の本質を「計算」として抽象しようとするものである。僕は、基本的には、「計算主義」の立場に立っている。

「知能」に対する「計算主義的」なアプローチは、僕が人工知能が将来実現すべき課題として関心を持っている、人間の「言語能力」や日常的な「論理的推論能力」、抽象的な「数学的能力」、さらには「科学する能力」の機械による実現には、そうしたアプローチが不可欠だと考えている。

この時、「感情」や「意識」等々の機械による知能実現にまつわる、様々な問題を、当面は括弧に入れて保留しているのだが。

もちろん、人間の「知能」にまつわる問題を、一つの方法論で対応するのは難しい。それについては、次の「知能の階層性」で、述べようと思う。

計算主義の含意

- 「計算」を、ある入力をある出力に変換する「機能」であると考えれば、同一性を残したまま、その具体的実装は捨象できる。ある計算が、そろばんの上でなされようと、スパコンの上でなされようと、はたまた、「計算」を担うのが、タンパク質だろうがシリコンだろうが、していることは「同じ」であると考えることはできるのである。
- 人間の「知能」の本質を「計算」として抽象することで、その抽象の限りでは、「機械」と「人間」は等値されうる。だから、この抽象のもとでは、機械ができることは人間にもできることになり、人間ができることは機械にもできることになる。特に、後者の「人間ができることは機械にもできる」という主張は、人工知能技術の可能性を考える上では、とても強い主張になる。人工知能の可能性を考える上では、「我々自身を見よ！」という以上の強いメッセージを、僕は、思いつかない。

計算主義と複雑性の理論

「それじゃ、逆に、Googleの巨大なCPUパワーで出来ることも、人間が手計算で出来ることも同じだということ？」

そう。同じなのだ。両者が、原理的には、紙テープを1マスずつ読み書きするTuringマシンと同値だという意味では。（もっとも、TuringマシンによるGoogleクラウドのシミュレーションが「多項式時間」で出来るか否かが、重要なポイントになるのだが。それについては、あとで「拡張されたチャーチ＝チューリング・テーゼ」のところで述べる。）

「計算」というの能力での人間と機械の同一性が、「計算主義」の要になる概念であるなら、計算能力の比較可能性をきちんと考える必要がある。そこに、「複雑性理論」が登場する。

「複雑性理論」というのは、「計算」の「複雑さ」を階層的に整理しようとする数学の一分野である。

複雑性理論

ただ、残念なことに、「複雑性理論」、まだまだわからないことが山のようにある。有名なのは、「P = NP ?」問題が、解けていないことである。研究の現状をまとめたサイトがここにある。

"Complexity Zoo" https://goo.gl/eabDas

なんと、「複雑さ」のカテゴリーが、500以上ある！

それは、「計算する知能」の構造が、予想以上に複雑だということを反映しているからだと僕は考えている。

我々は、我々自身のことをよく知らないのである。

ただ、僕は、この複雑性理論が、「計算主義」の立場から人工知能技術の可能性を考える上で、もっとも基本的な手段を提供してくれると考えている。

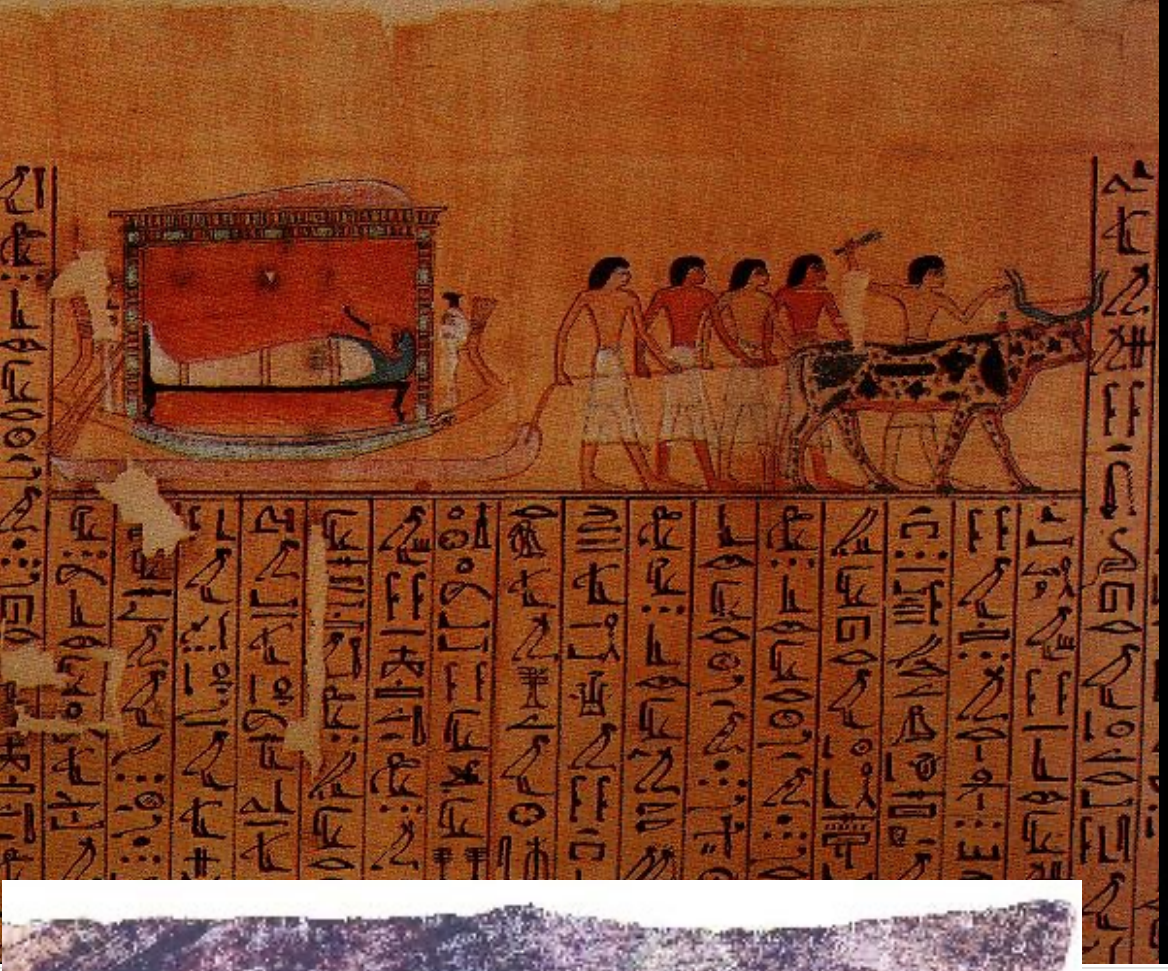
「知能」の階層性

知能の階層性

- 人間の知能は、生物学的進化の過程で獲得した、他の動物とも共通する「感覚＝運動的」な認識能力の上に、いくつかの階層が積み重なって構成されている。
- 他の動物と人間とを区別する、人間という種に固有の能力である言語能力は、生物学的な先天的で生得的な能力を基礎としている。
- 生物学的進化の過程とは区別できる、人間の社会史の発展の中で形成された、認識能力の階層を人間は持っている。
  その層は、いろいろな区分が可能なのであるが、ここでは、「文字による知識の集積」と「数学的・科学的な認識」をピックアップしている。（「技術的知」というカテゴリーも可能かと思うのだが。）


アノマロカリス
カンブリア紀：視覚の獲得


人類の言語能力の獲得




文字







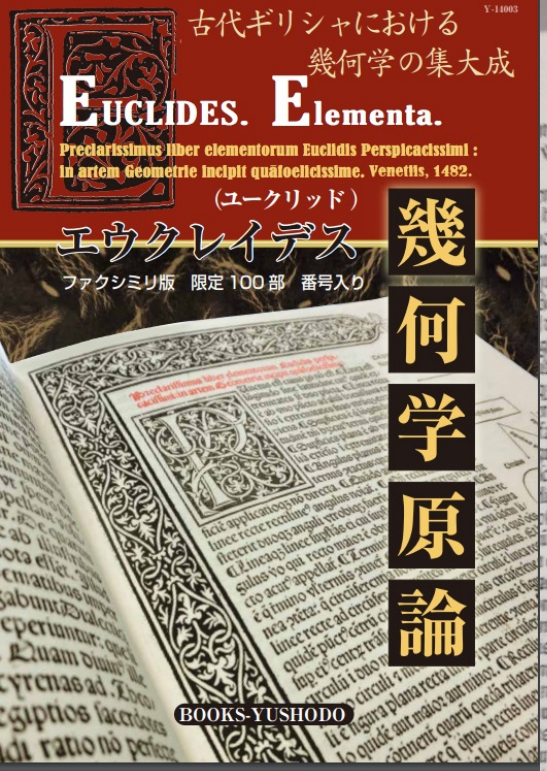
バビロニアの数学　Plimpton 322

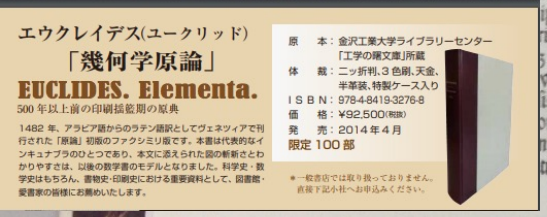
バビロニアでは、ピタゴラスの定理を知っていた。



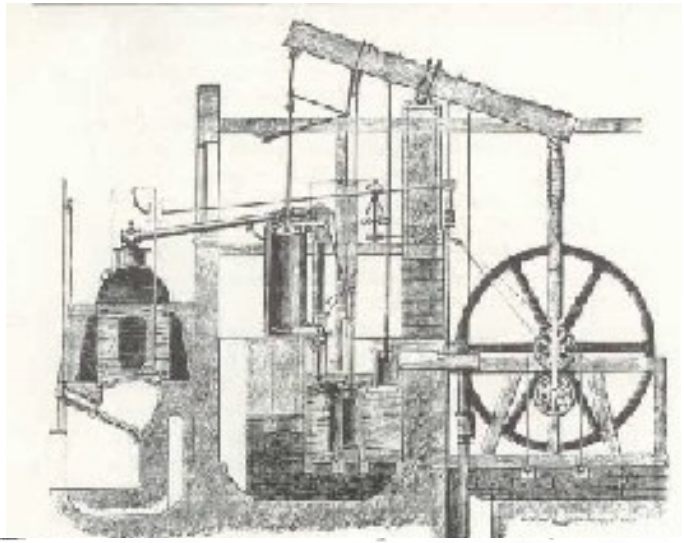
1822 BC ~ 1762 BC

印刷術

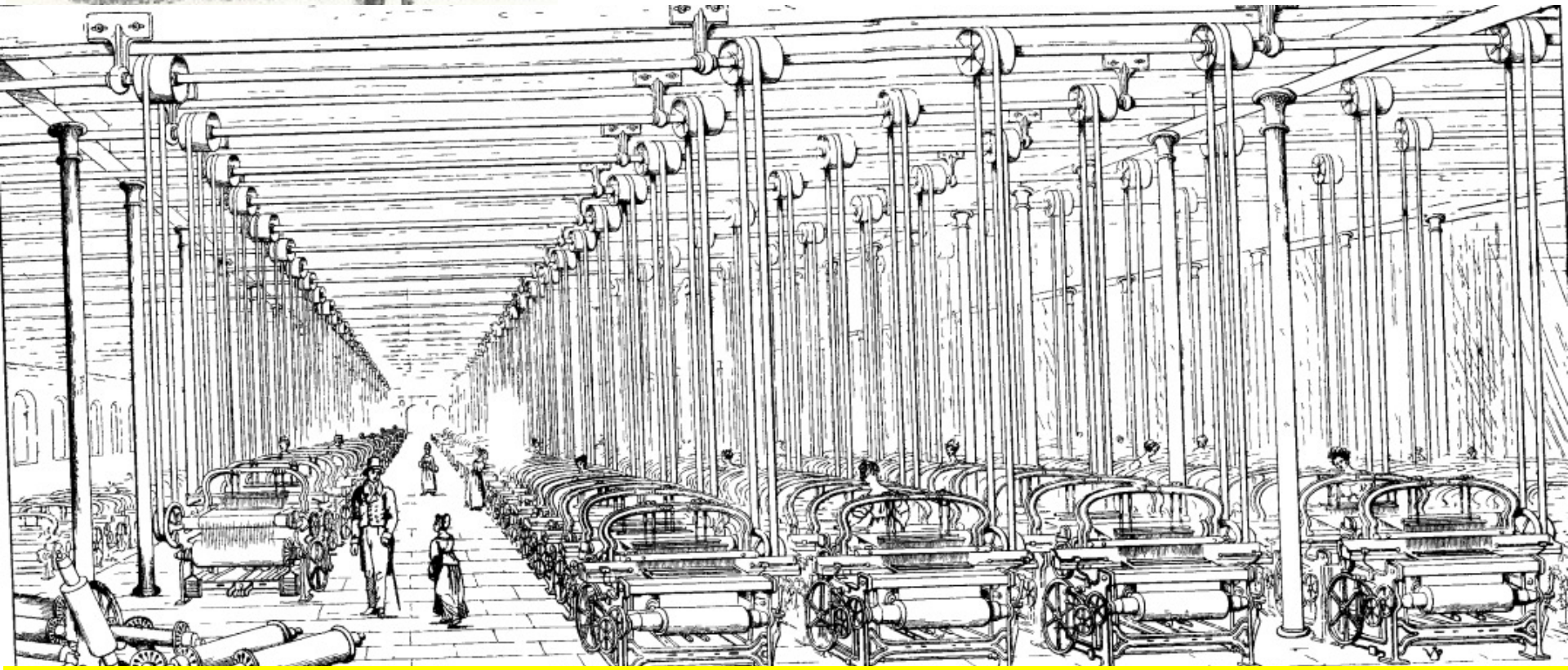




1482 年、アラビア語からのラテン語訳としてヴェネツィアで刊行された「原論」初版のファクシミリ版



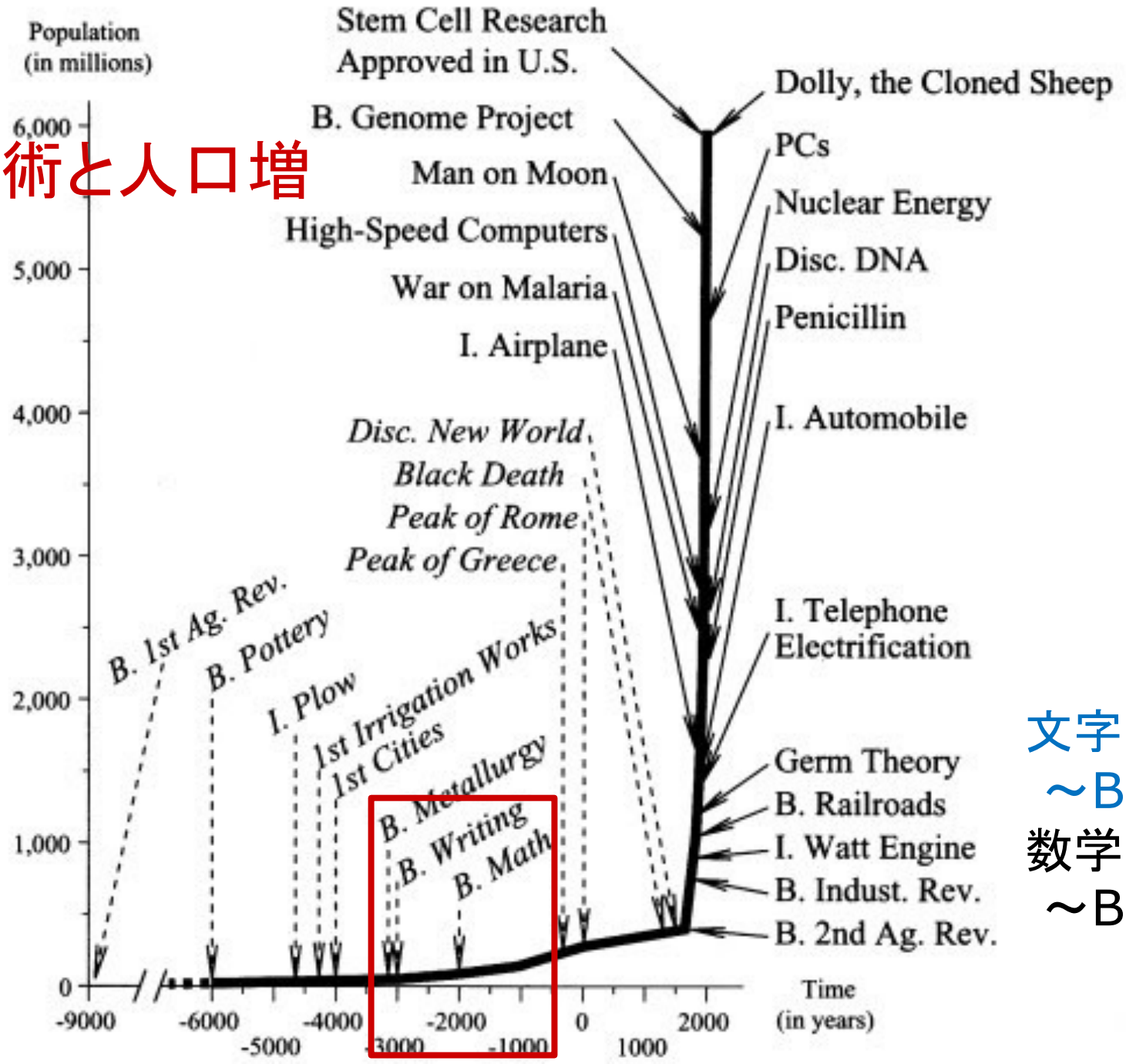
機械制大工業



http://socserv.mcmaster.ca/econ/ugcm/3ll3/ure/PhilosophyManufactures.pdf

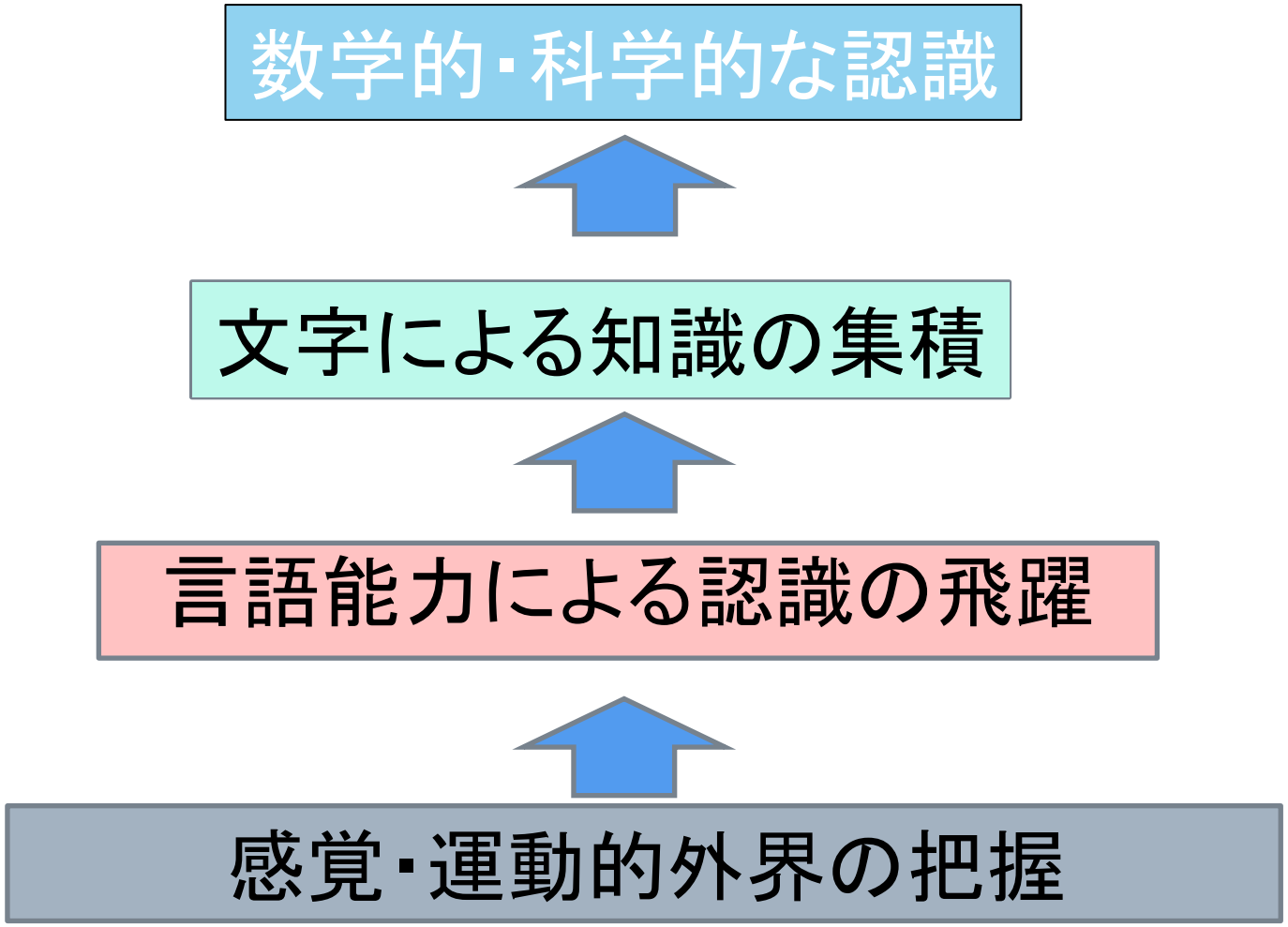

Foxconn iPhone組み立て工場

技術と人口増

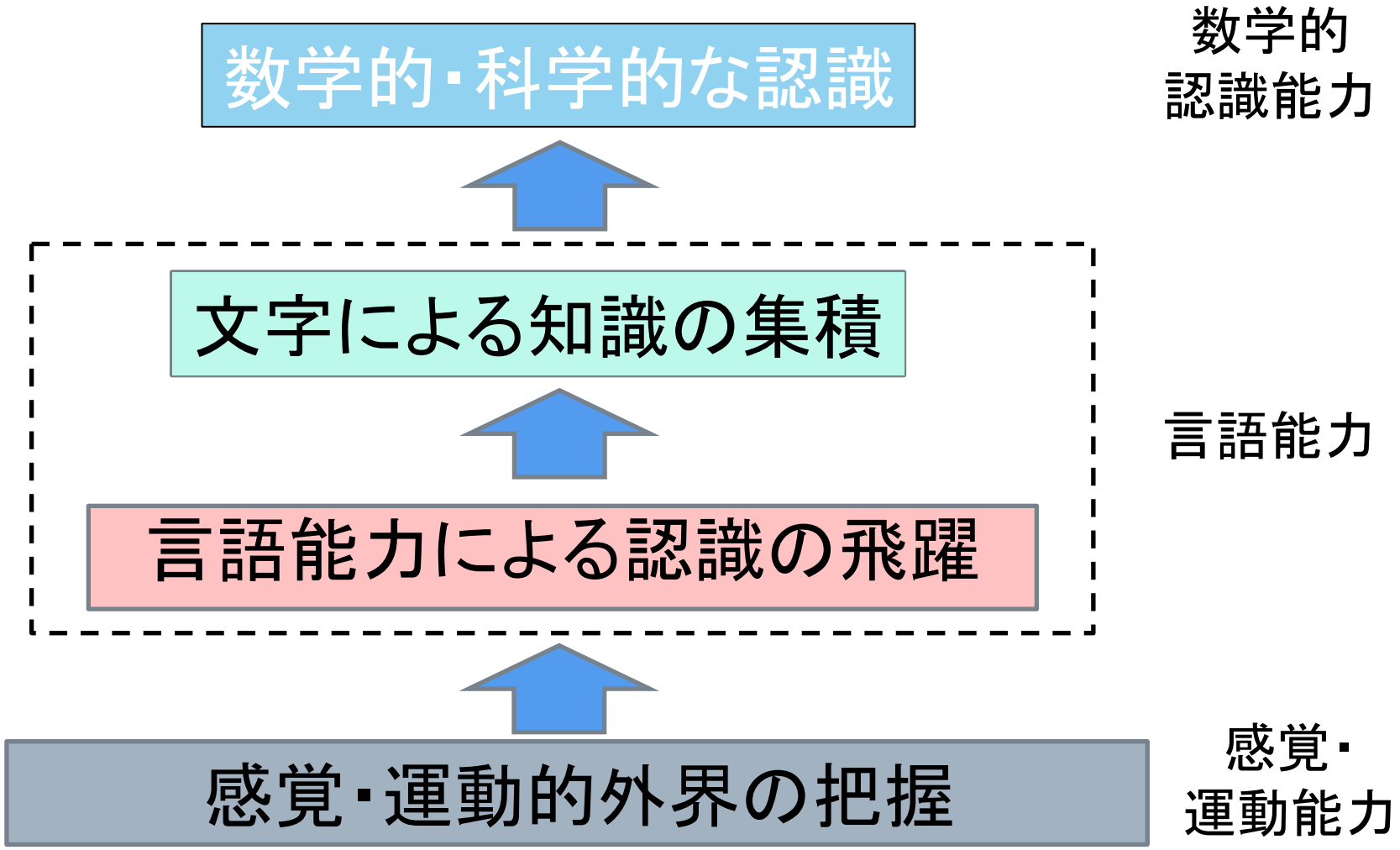


文字の誕生
～BC3000年
数学の誕生
～BC2000年

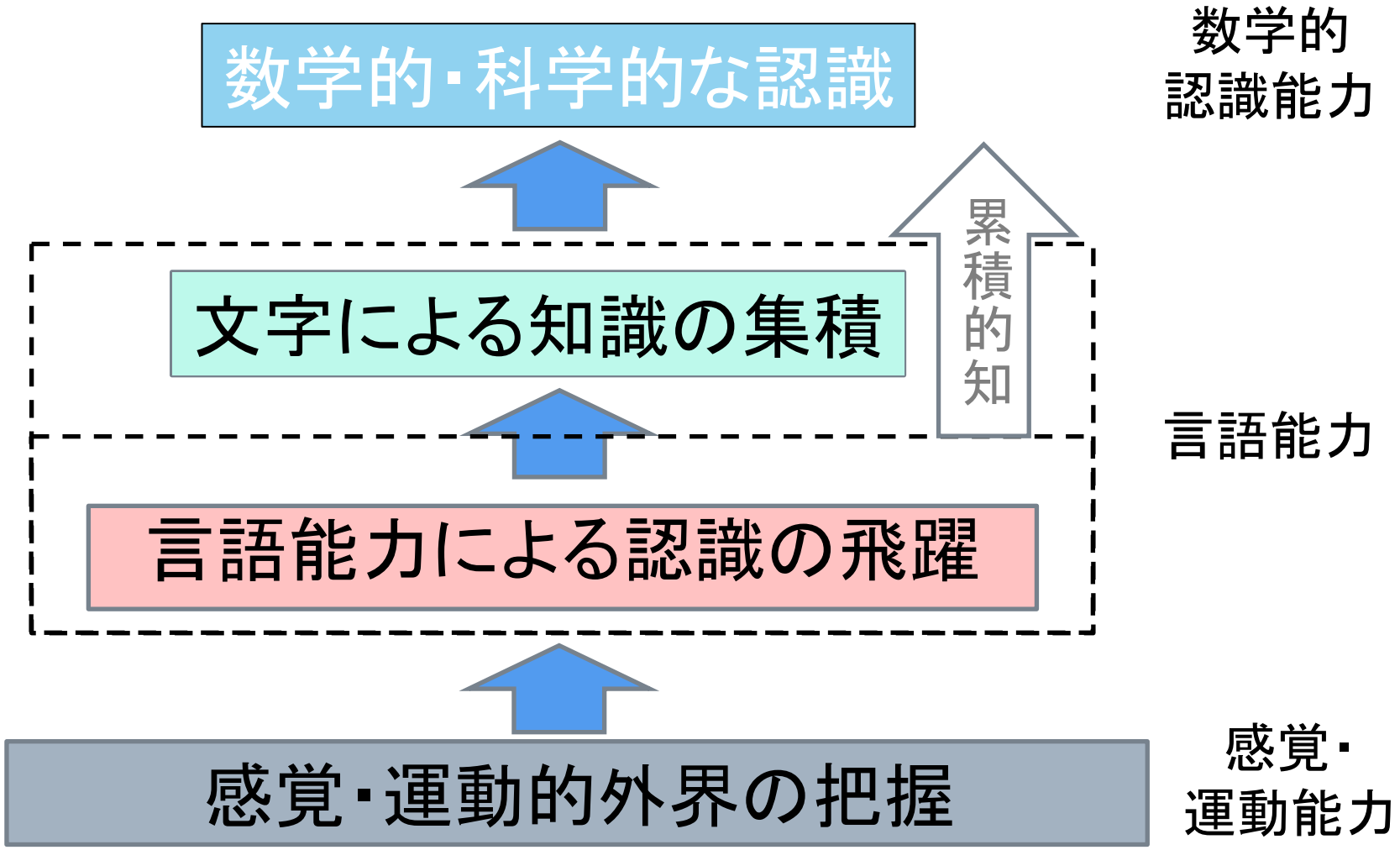
認識能力の発展と人間の知能の階層



認識能力の発展と人間の知能の階層

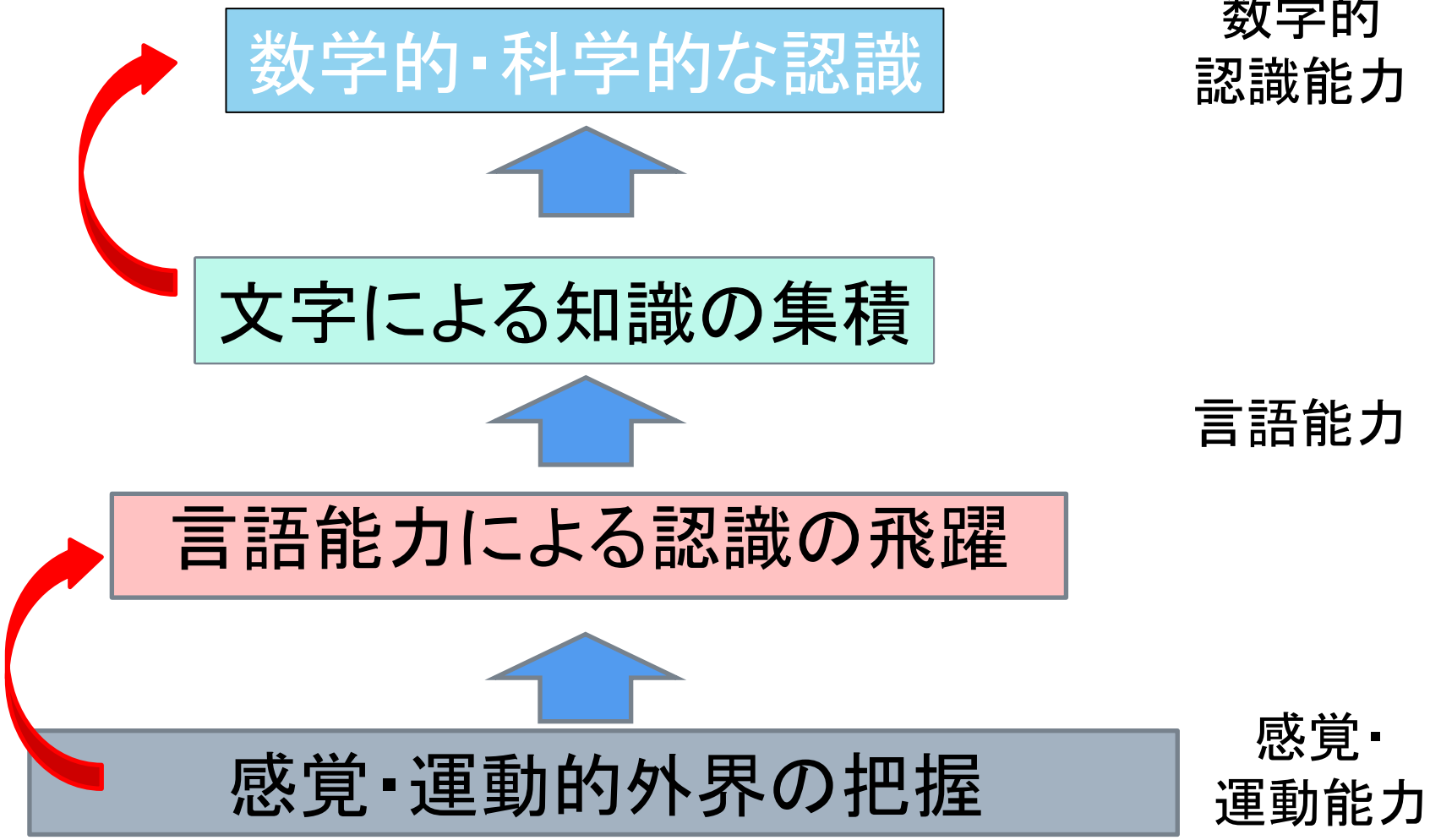


認識能力の発展と人間の知能の階層

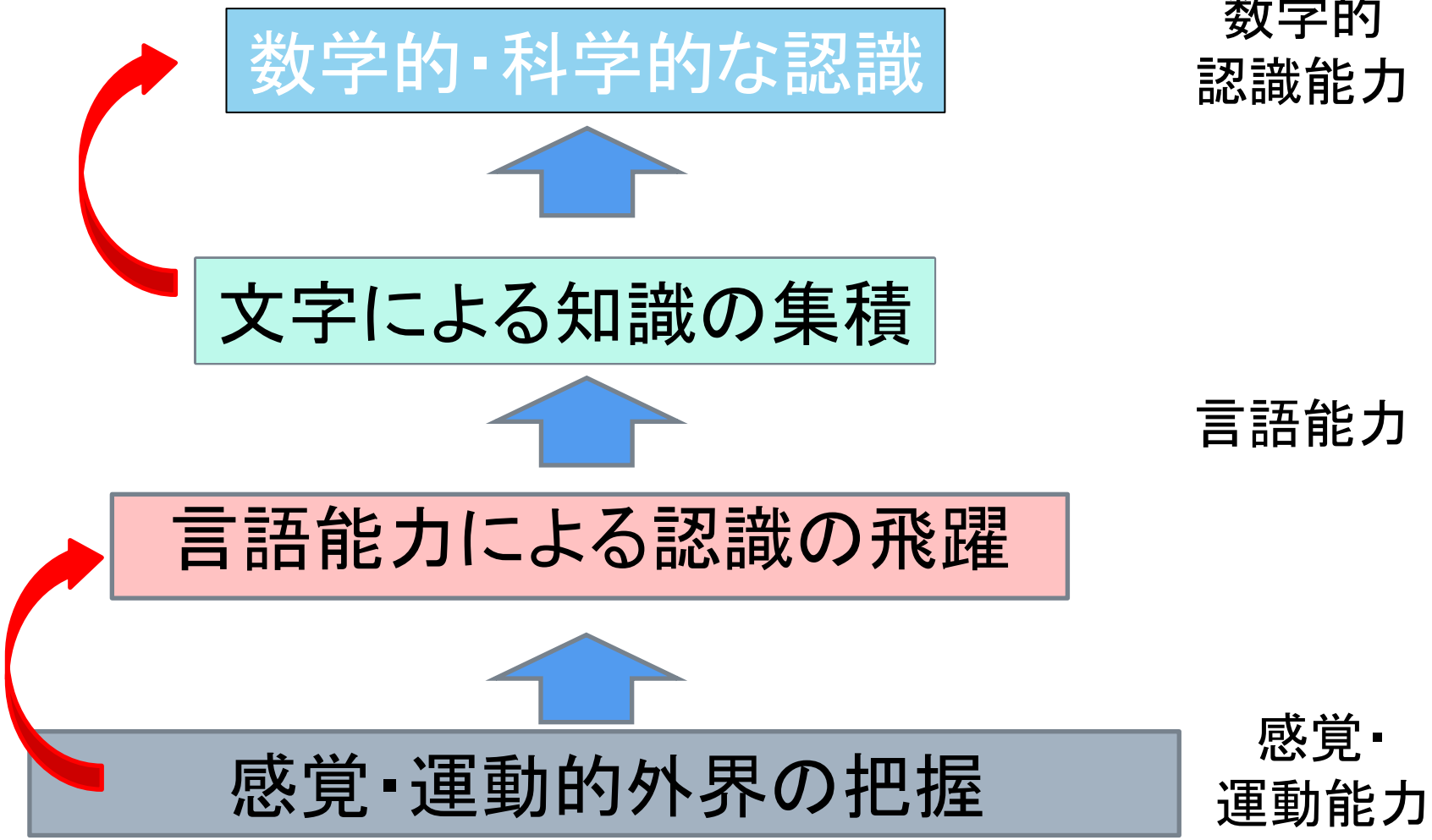


認識能力の発展と人間の知能の階層

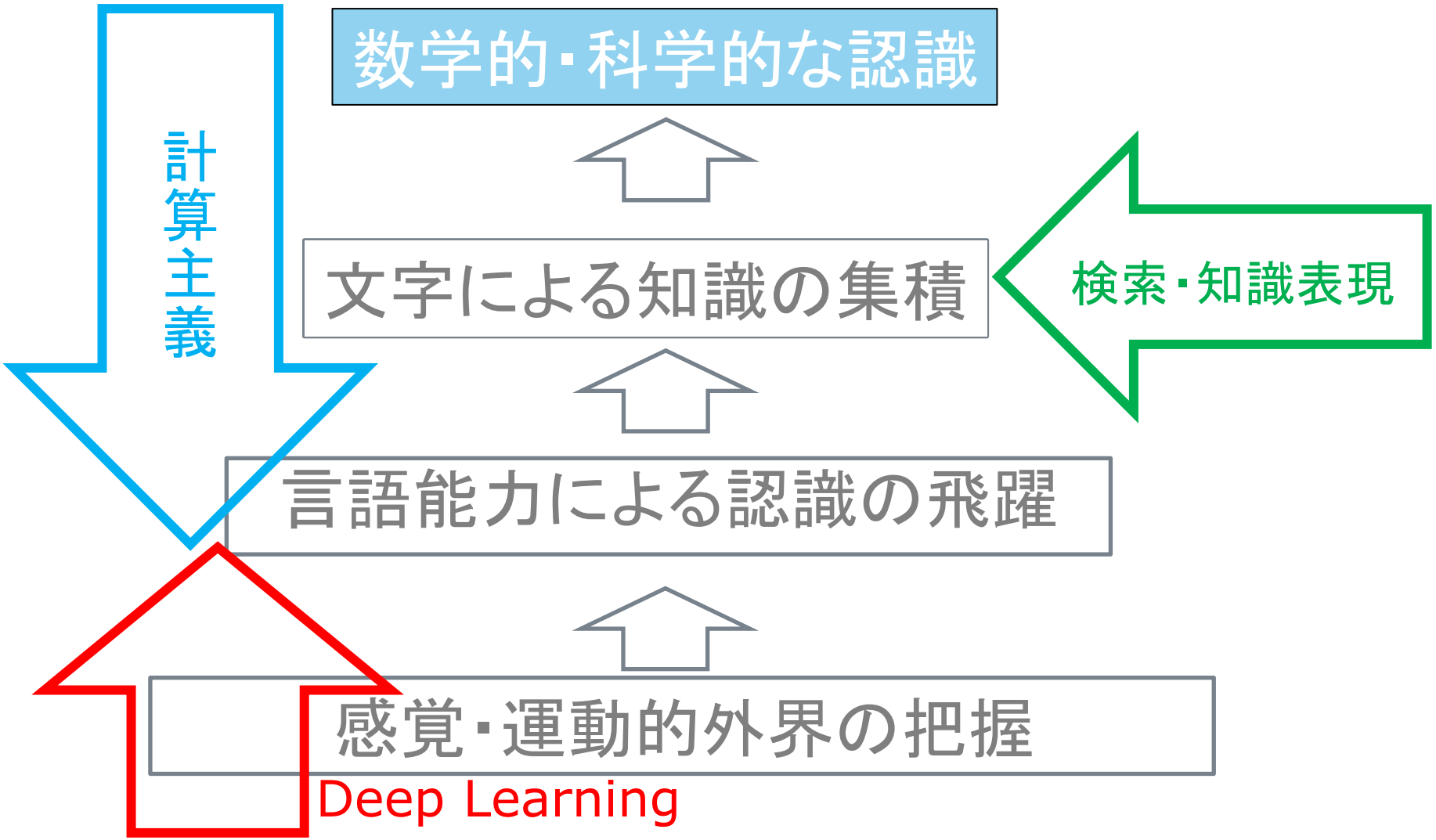
大きな飛躍は二つある



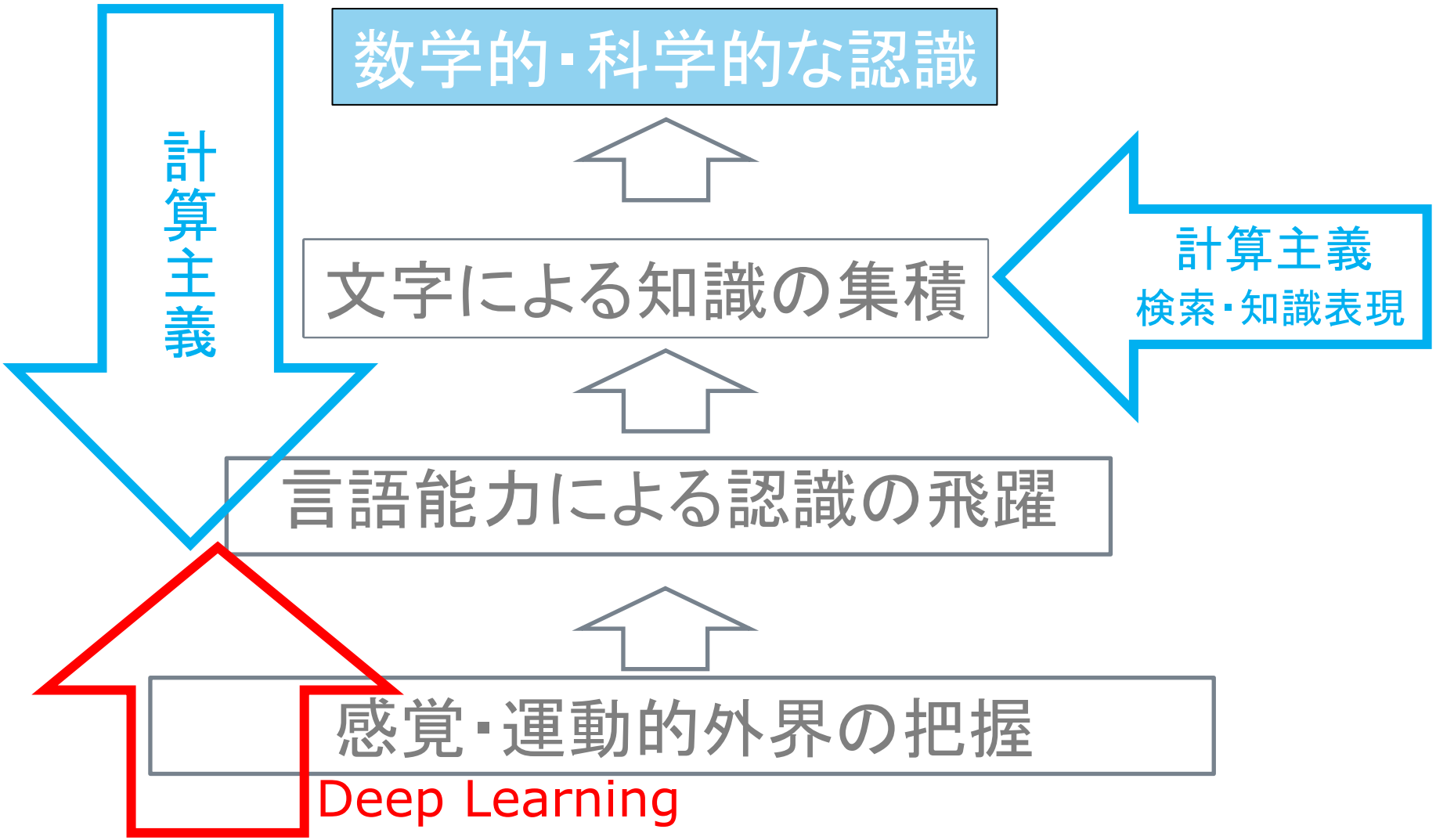
そして、それは現在の人工知能技術が攻めあぐねている課題である



複数のアプローチが必要



計算主義の重要性は高まる



コネクショニズムの
「形式性」の認識能力と「生成能力」

ディープラーニングでできること

- ここでは、ディープラーニングの手法を通じて、データの中に「形式的特徴」を見出すことができることを示そうと思う。
- また、見つけ出した特徴に基づいて、そうした特徴を持つ、新しいデータを生成できることも示そうと思う。
- ただ、こうしたアプローチには、いくつかの問題があることも、明らかにしようと思う。

- 基本的には、そこで獲得された「形式的知」は、あるシステムとそこでのデータに固有の個別的・具体的知として、システムとデータに強く結びついていて、一般的・抽象的な「形式的知」としては、そこから分離できない。（言語の話し手が、母語の文法に正確に従っていながら、必ずしも、その文法を正確に記述できないのと似ている。） それは「知の累積性」を担保できない。

LSTMの能力について -- 原論文を読む

"Long Short Term Memory"

Sepp Hochreiter et al.

https://goo.gl/sDAq81

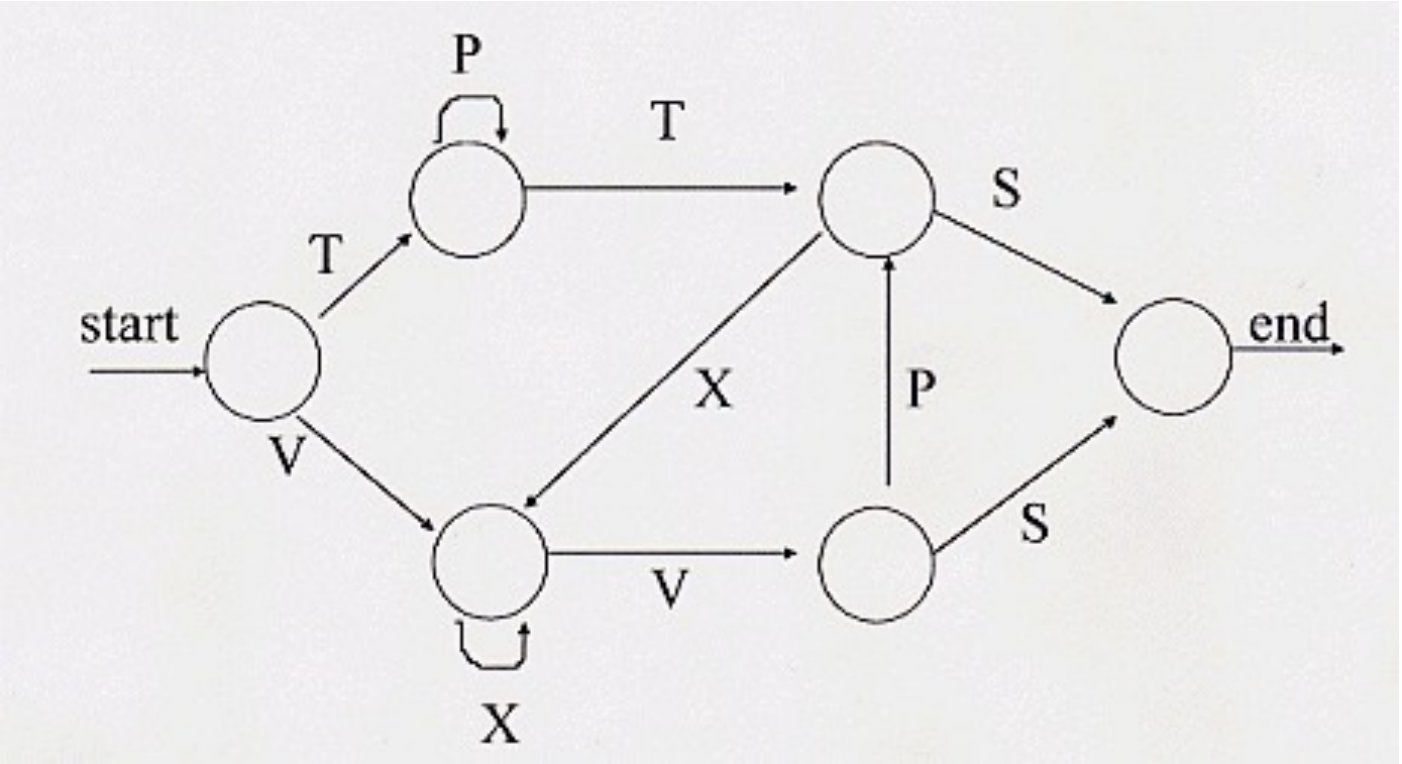
1997年



Hochreiterの90年代の発見

- LSTMの提案者であるHochreiterは、1990年代に、既に RNNが、極めて高い、認識能力を持つことを発見していた。LSTMの原論文に、詳細に、多数の実験結果が報告されている。ここでは、その中から、三つほどの例を紹介する。
- 一つは、有限オートマトンの生成する文字列の規則性を、LSTMが認識できるという実験結果である。これは、認知心理学の分野で、人間が持つ「人工的文法の学習（Artificial grammar learning）」能力として研究されていたものである。 https://goo.gl/VgvboK
- もう一つは、LSTMが、掛け算を学習できるという実験である。ニューラルネットが、任意の関数の近似が可能であることは知られていたのだが、これはこれで興味ふかい実験である。
- 最後は、あるルールに従って、時間的に離れたところで起きるイベントのパターンを、LSTMが学習できるという実験である。これは、人間でも手こずりそうかもしれない。
- いずれの実験も、膨大な学習の繰り返しが必要である。

人工的な文法の学習
実験 1: REBER GRAMMAR



例えば、次の文字列、**VXVS, TPTXVS** は、文法にあっているが、
VXXXS, TPTPS は文法的ではない。

実験 1: Embedded Reber grammar

LSTMは、この文法を学習できる！

method	hidden units	# weights	learning rate	% of success	success after
RTRL	3	≈ 170	0.05	"some fraction"	173,000
RTRL	12	≈ 494	0.1	"some fraction"	25,000
ELM	15	≈ 435		0	>200,000
RCC	7-9	≈ 119-198		50	182,000
LSTM	4 blocks, size 1	264	0.1	100	39,740
LSTM	3 blocks, size 2	276	0.1	100	21,730
LSTM	3 blocks, size 2	276	0.2	97	14,060
LSTM	4 blocks, size 1	264	0.5	97	9,500
LSTM	3 blocks, size 2	276	0.5	100	8,440

掛け算の学習
実験5: MULTIPLICATION PROBLEM

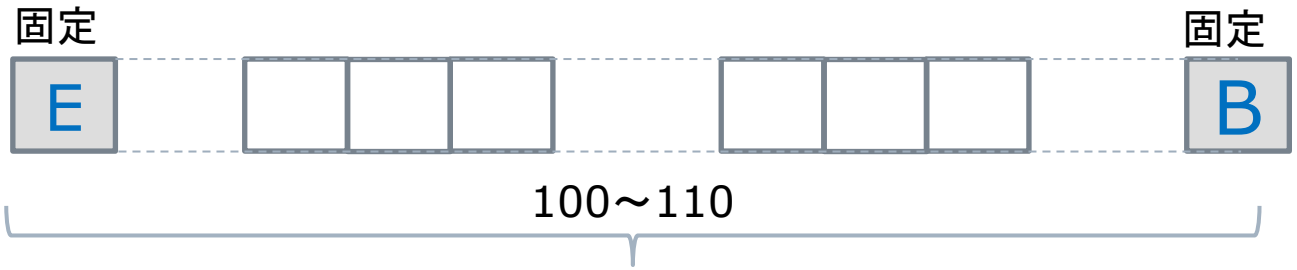
- **Task**. Like the task in Section 5.4, except that the first component of each pair is a real value randomly chosen from the interval [0; 1]. In the rare case where the rst pair of the input sequence gets marked, we set X1 to 1.0. The target at sequence end is the product X1 x X2.

T	minimal lag	# weights	n_{seq}	# wrong predictions	MSE	Success after
100	50	93	140	139 out of 2560	0.0223	482,000
100	50	93	13	14 out of 2560	0.0139	1,273,000

LSTMは、足し算（実験4）も掛け算も学習できる！

実験6:TEMPORAL ORDER

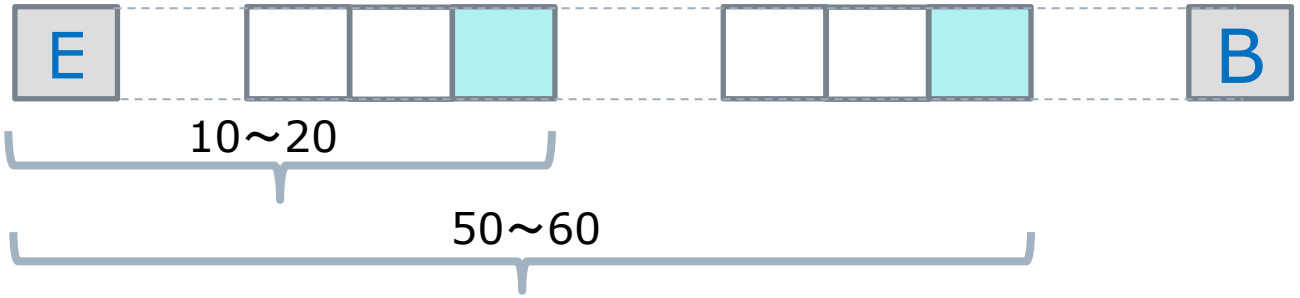
先頭にE、最後にBが入っている（固定）、長さ100～110（可変）の文字列がある。



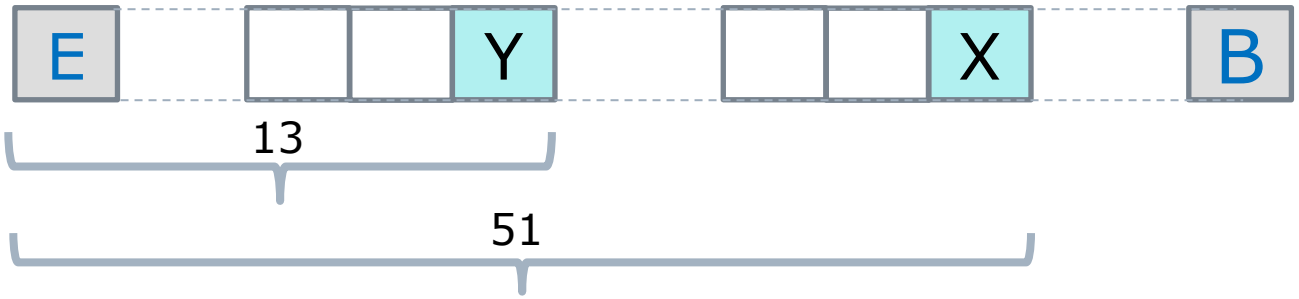
先頭のE、最後のB以外の場所には、a, b, c, d の文字がランダムに入っている。

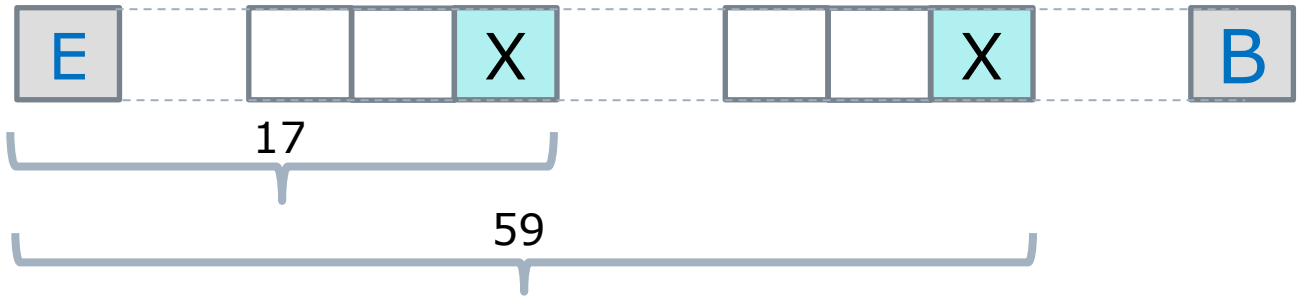


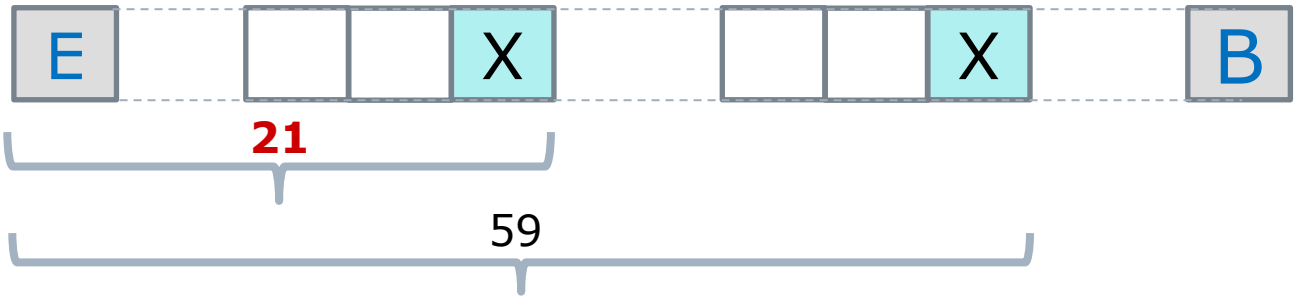
ただし、先頭から10～20番目と50～60番目のランダムに選ばれたな二箇所に、XまたはYの文字が入っている。



例えば、次の二例は、条件を満たすが、三番目の例は、条件を満たしていない。







この時、範囲指定された二箇所に、

X,Xが入っているならQ、
X,Yが入っているならR、
Y,Xが入っているならS、
Y,Yが入っているならU

として、与えられた文字列を分類する。

先の第一例はS、第二例はQ、
第三例は、そのいずれでもないことになる。

task	# weights	# wrong predictions	Success after
Task 6a	156	1 out of 2560	31,390
Task 6b	308	2 out of 2560	571,100

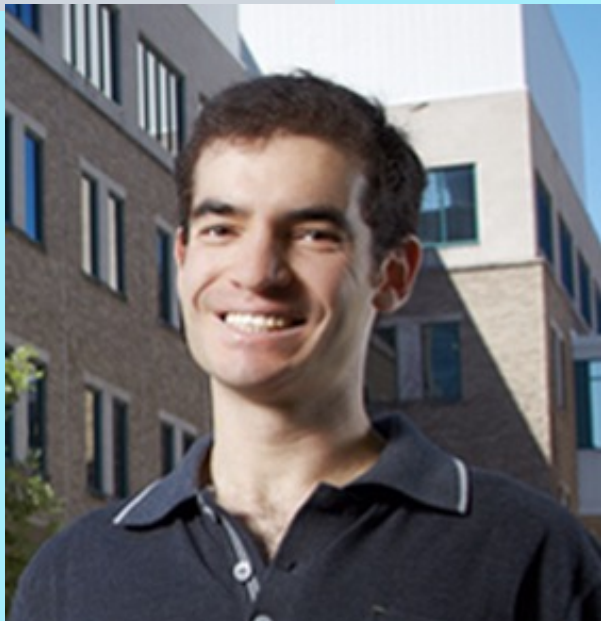
Task6bは、三箇所版。
X,X,X -> Q; X,X,Y -> R; X,Y, X -> S; X,Y,Y -> U;
Y,X,X -> V ; Y,X,Y -> A; Y,Y,X -> B; Y,Y,Y -> C
の8分類。

RNNによる文の生成

"Generating Text with Recurrent Neural Networks"

Ilya Sutskever et al.
http://goo.gl/vHRHSn

2011年



ニューラルネットによる文章の生成

- Googleの Ilya Sutskeverは、文字数が5億文字にものぼるテキストを長い時間をかけてRecurrent Neural Nets に学習させ、次のページのような文章を生成することができた。

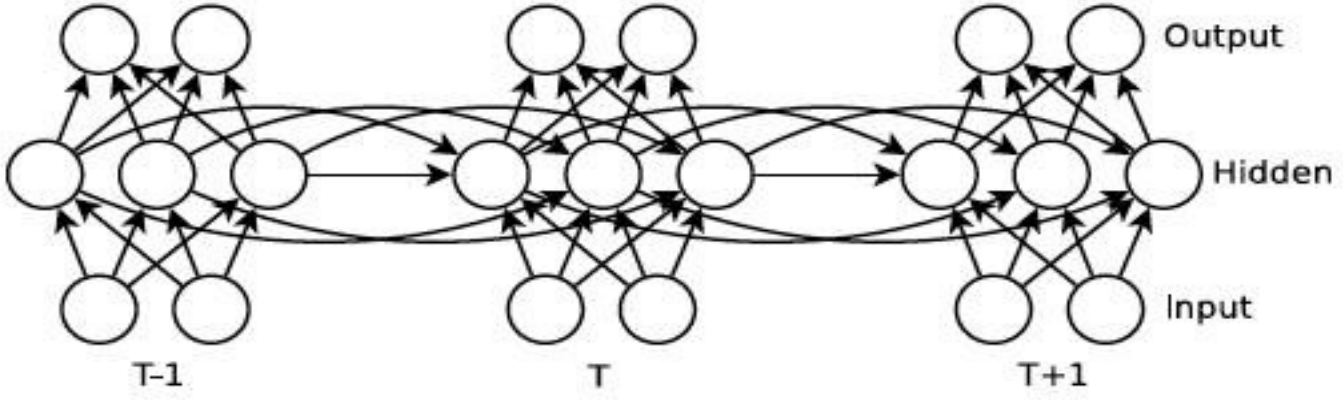


Figure 1. A Recurrent Neural Network is a very deep feedforward neural network whose weights are shared across time. The non-linear activation function used by the hidden units is the source of the RNN's rich dynamics.

“An example of what recurrent neural nets can now do” WikiPedia で学習したもの

- The meaning of life is the tradition of the ancient human reproduction: it is less favorable to the good boy for when to remove her bigger. In the show’s agreement unanimously resurfaced. The wild pasteured with consistent street forests were incorporated by the 15th century BE. In 1996 the primary rapford undergoes an effort that the reserve conditioning, written into Jewish cities, sleepers to incorporate the .St Eurasia that activates the population.

http://goo.gl/vHRHSn

“An example of what recurrent neural nets can now do” New York Timesで学習

- while he was giving attention to the second advantage of school building a 2-for-2 stool killed by the Cultures saddled with a halfsuit defending the Bharatiya Fernall ’s office . Ms . Claire Parters will also have a history temple for him to raise jobs until naked Prodiena to paint baseball partners , provided people to ride both of Manhattan in 1978 , but what was largely directed to China in 1946 , focusing on the trademark period is the sailboat yesterday and comments on whom they obtain overheard within the 120th anniversary , where .....

http://goo.gl/vHRHSn

Colorless green ideas sleep furiously

is a sentence composed by Noam Chomsky in his 1957 book *Syntactic Structures* as an example of a sentence that is grammatically correct, but semantically nonsensical.

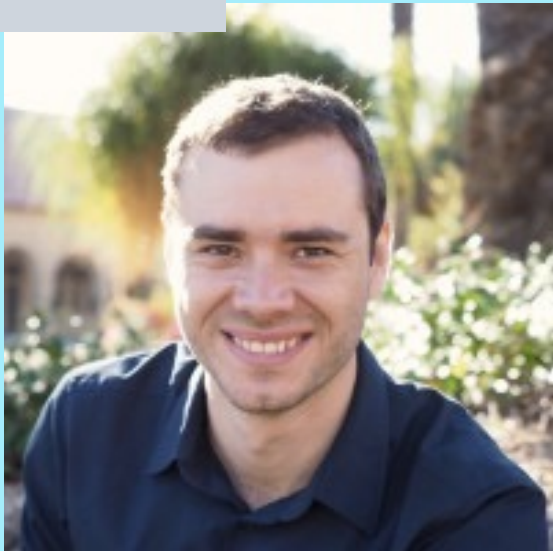
RNNの驚くべき能力

"The Unreasonable Effectiveness of Recurrent Neural Networks"

Andrej Karpathy

http://goo.gl/mNqwCv

2015年



```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
 int error;
 if (fd == MARN_EPT) {
  /*
   * The kernel blank will coeld it to userspace.
   */
  if (ss->segment < mem_total)
   unblock_graph_and_set_blocked();
  else
   ret = 1;
  goto bail;
 }
 segaddr = in_SB(in.addr);
 selector = seg / 16;
 setup_works = true;
 for (i = 0; i < blocks; i++) {
  seq = buf[i++];
  bpf = bd->bd.next + i * search;
  if (fd) {
   current = blocked;
  }
```

RNNが産み出した
Cプログラム・モドキ

Linuxのソースコードを
「学習」させたもの

```
/*
 * If this error is set, we will need anything right after that BSD.
 */
static void action_new_function(struct s_stat_info *wb)
{
  unsigned long flags;
  int lel_idx_bit = e->edd, *sys & ~((unsigned long) *FIRST_COMPAT);
  buf[0] = 0xFFFFFFFF & (bit << 4);
  min(inc, slist->bytes);
  printk(KERN_WARNING "Memory allocated %02x/%02x, "
    "original MLL instead\n"),
    min(min(multi_run - s->len, max) * num_data_in),
    frame_pos, sz + first_seg);
  div_u64_w(val, inb_p);
  spin_unlock(&disk->queue_lock);
  mutex_unlock(&s->sock->mutex);
  mutex_unlock(&func->mutex);
  return disassemble(info->pending_bh);
}

static void num_serial_settings(struct tty_struct *tty)
{
  if (tty == tty)
    disable_single_st_p(dev);
  pci_disable_spool(port);
  return 0;
}
```

RNNが産み出した
Cプログラム・モドキ

Linuxのソースコードを
「学習」させたもの

For $\bigoplus_{n=1,\ldots,m}$ where $\mathcal{L}_{m\bullet} = 0$, hence we can find a closed subset $\mathcal{H}$ in $\mathcal{H}$ and any sets $\mathcal{F}$ on X, U is a closed immersion of S, then $U \to T$ is a separated algebraic space.

Proof. Proof of (1). It also start we get

$$S = \mathrm{Spec}(R) = U \times_X U \times_X U$$

and the compariciy in the fibre product covering we have to prove the lemma generated by $\coprod Z \times_U U \to V$. Consider the maps M along the set of points Sch_{fppf} and $U \to U$ is the fibre category of S in U in Section, ?? and the fact that any U affine, see Morphisms, Lemma ??. Hence we obtain a scheme S and any open subset $W \subset U$ in $Sh(G)$ such that $\mathrm{Spec}(R') \to S$ is smooth or an

$$U = \bigcup U_i \times_{S_i} U_i$$

which has a nonzero morphism we may assume that f_i is of finite presentation over S. We claim that $\mathcal{O}_{X,x}$ is a scheme where $x, x', s'' \in S'$ such that $\mathcal{O}_{X,x'} \to \mathcal{O}'_{X',x'}$ is separated. By Algebra, Lemma ?? we can define a map of complexes $\mathrm{GL}_{S'}(x'/S'')$ and we win. □

To prove study we see that $\mathcal{F}|_U$ is a covering of $\mathcal{X}'$, and $\mathcal{T}_i$ is an object of $\mathcal{F}_{X/S}$ for $i > 0$ and $\mathcal{F}_p$ exists and let $\mathcal{F}_i$ be a presheaf of $\mathcal{O}_X$-modules on $\mathcal{C}$ as a $\mathcal{F}$-module. In particular $\mathcal{F} = U/\mathcal{F}$ we have to show that

$$\widetilde{M}^\bullet = \mathcal{I}^\bullet \otimes_{\mathrm{Spec}(k)} \mathcal{O}_{S,s} - i_X^{-1}\mathcal{F})$$

is a unique morphism of algebraic stacks. Note that

$$\mathrm{Arrows} = (Sch/S)^{opp}_{fppf}, (Sch/S)_{fppf}$$

and

$$V = \Gamma(S, \mathcal{O}) \longmapsto (U, \mathrm{Spec}(A))$$

is an open subset of X. Thus U is affine. This is a continuous map of X is the inverse, the groupoid scheme S.

Proof. See discussion of sheaves of sets.

RNNが産み出した数学論文モドキ

Stack Theory の教科書を「学習」させたもの

Proof. Omitted. □

Lemma 0.1. *Let $\mathcal{C}$ be a set of the construction.*

Let $\mathcal{C}$ be a gerber covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $X_{\acute{e}tale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules. □

Lemma 0.2. *This is an integer $\mathcal{Z}$ is injective.*

Proof. See Spaces, Lemma ??. □

Lemma 0.3. *Let S be a scheme. Let X be a scheme and X is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let X be a scheme. Let X be a scheme which is equal to the formal complex.*

The following to the construction of the lemma follows.

Let X be a scheme. Let X be a scheme covering. Let

$$b : X \to Y' \to Y \to Y \to Y' \times_X Y \to X.$$

be a morphism of algebraic spaces over S and Y.

Proof. Let X be a nonzero scheme of X. Let X be an algebraic quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

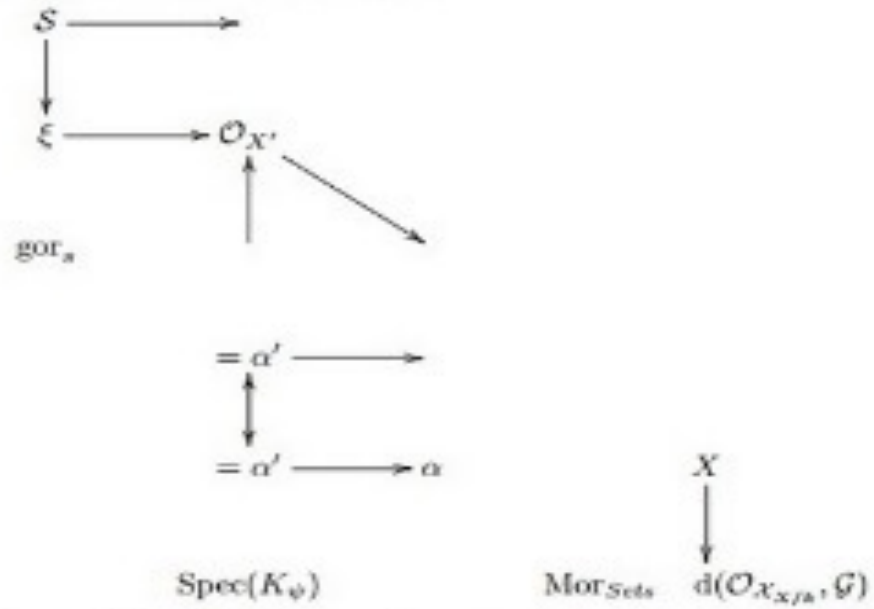
(1) $\mathcal{F}$ is an algebraic space over S.
(2) If X is an affine open covering.

Consider a common structure on X and X the functor $\mathcal{O}_X(U)$ finite type.

RNNが産み出した数学論文モドキ

Stack Theory の教科書を「学習」させたもの

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram



is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

□

Proof. We have see that $X = \mathrm{Spec}(R)$ and $\mathcal{F}$ is a finite type representable by algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then the cohomology of X is an open neighbourhood of U.

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas ??.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The fu
"field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad \text{-1}(\mathcal{O}_{X_{\acute{e}tale}}) \longrightarrow \mathcal{O}_{X_i}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\nu}^{\overline{x}})$$

is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type o
If $\mathcal{F}$ is a scheme theoretic image points.

If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma ??
sequence of $\mathcal{F}$ is a similar morphism.

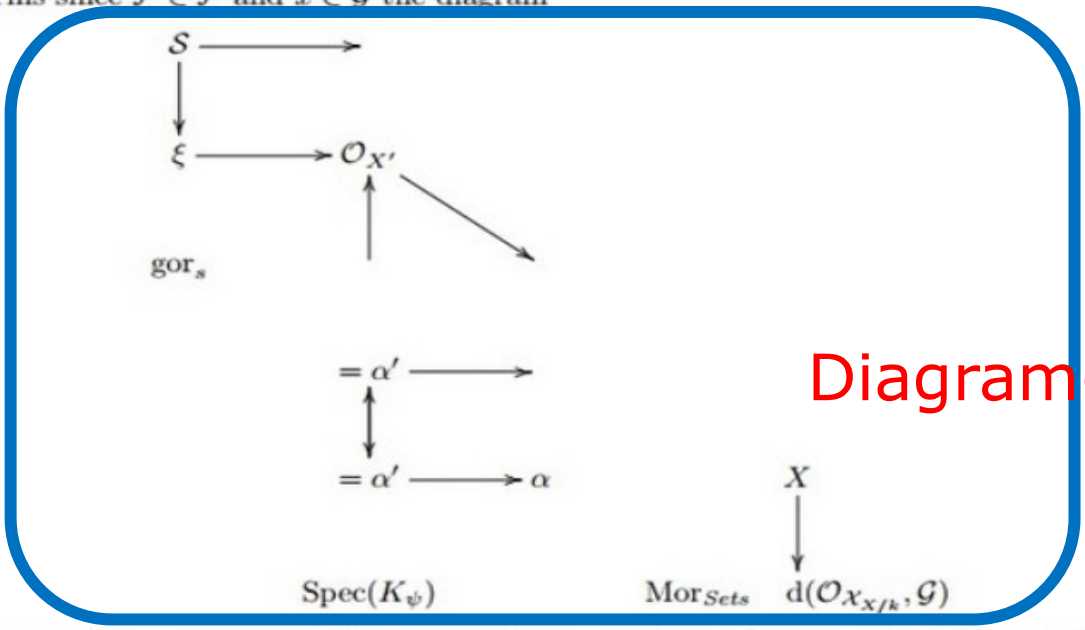
RNNが産み出した
数学論文モドキ

Stack Theory の教科書を
「学習」させたもの

RNNは、Latexの構文規則を学習し、
こうしたシーケンスを構成できる。

```
\begin{proof}
We may assume that $\mathcal{I}$ is an abelian sheaf on $\mathcal{C}$.
\item Given a morphism $\Delta : \mathcal{F} \to \mathcal{I}$
is an injective and let $\mathfrak q$ be an abelian sheaf on $X$.
Let $\mathcal{F}$ be a fibered complex. Let $\mathcal{F}$ be a category.
\begin{enumerate}
\item \hyperref[setain-construction-phantom]{Lemma}
\label{lemma-characterize-quasi-finite}
Let $\mathcal{F}$ be an abelian quasi-coherent sheaf on $\mathcal{C}$.
Let $\mathcal{F}$ be a coherent $\mathcal{O}_X$-module. Then
$\mathcal{F}$ is an abelian catenary over $\mathcal{C}$.
\item The following are equivalent
\begin{enumerate}
\item $\mathcal{F}$ is an $\mathcal{O}_X$-module.
\end{lemma}
```

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram



Diagramの真似ができない！

is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

Proof. We have see that $X = \mathrm{Spec}(R)$ and $\mathcal{F}$ is a finite type representable b algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then th cohomology of X is an open neighbourhood of U.

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas ??.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad -1(\mathcal{O}_{X_{\acute{e}tale}}) \longrightarrow \mathcal{O}_{X_\ell}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{v}})$$

is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

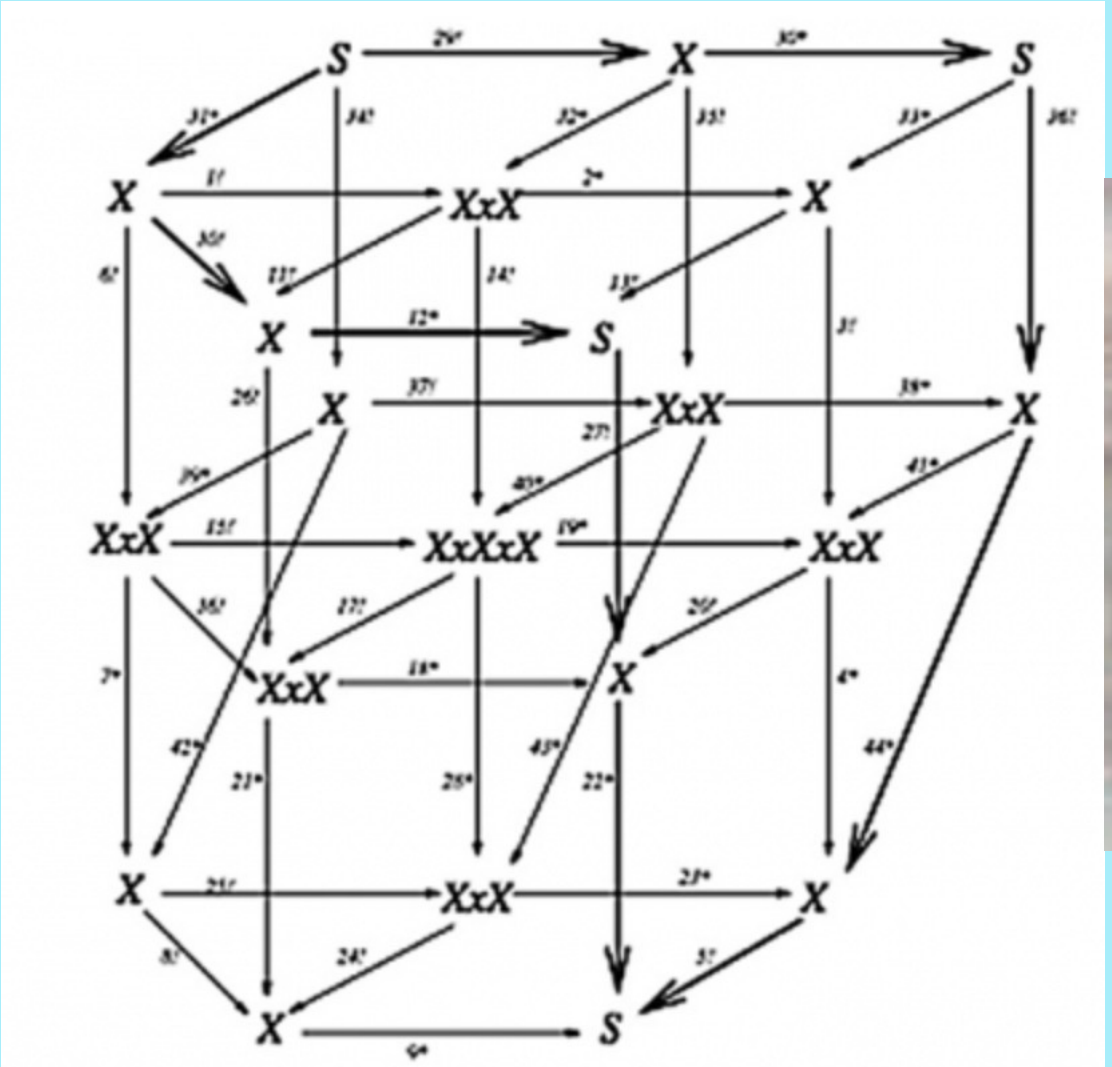
If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma ??. This is a sequence of $\mathcal{F}$ is a similar morphism.

RNNが産み出した数学論文モドキ

Stack Theory の教科書を「学習」させたもの

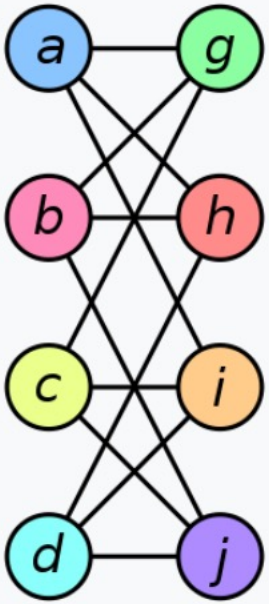
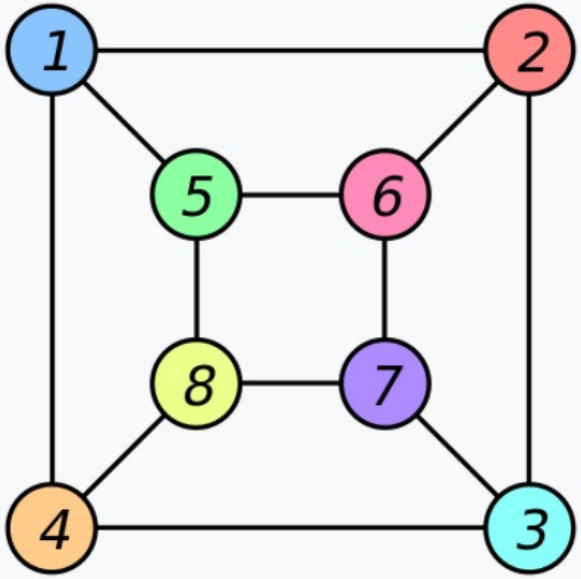
生成できない！

生成できる！



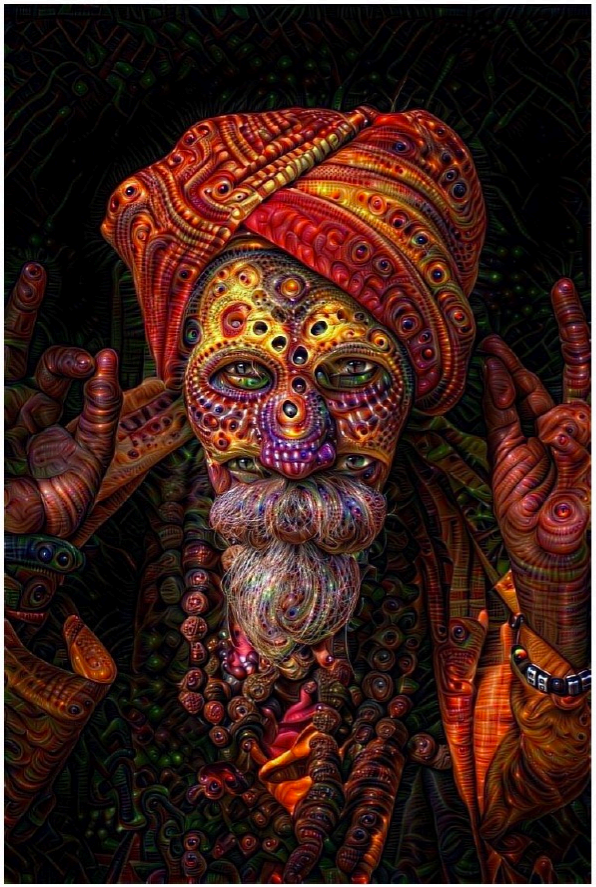


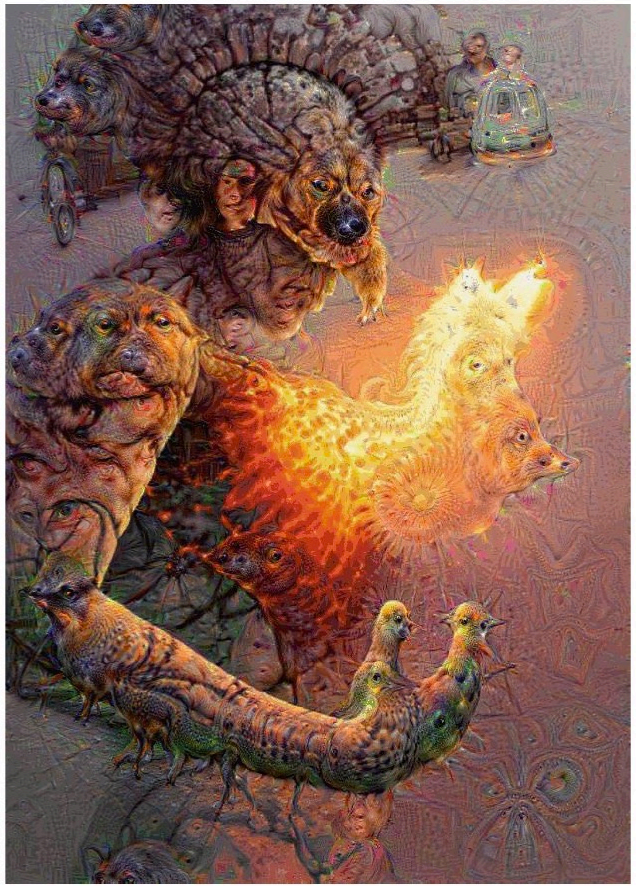
先の空白部分には、
こんな図形が入るはず

Graph G	Graph H	An isomorphism between G and H
		$f(a) = 1$ $f(b) = 6$ $f(c) = 8$ $f(d) = 3$ $f(g) = 5$ $f(h) = 2$ $f(i) = 4$ $f(j) = 7$

グラフ同型　　NP
部分グラフ同型　　NP-完全

生成できる！









reddit.com/r/deepdream

Part II

計算可能性理論

Agenda Part II
計算可能性理論

- 計算可能性理論
  - ゲーデルの不完全性定理
  - Church-Turing Thesis
- 計算主義の源流　TuringとChomsky
  - チューリングの問題提起
    「機械は考えることができるか？」
  - 文法の階層性　Chomsky Hierarchyについて
- 「有限の立場」と「実効的計算可能性」
  - 一般帰納関数の計算可能性　アッカーマン関数
  - Cantorの順序数
    - ゲンツェンの数論の無矛盾性証明

ゲーデルの不完全性定理

“On Formally Undecidable Propositions of Principia Mathematica and Related Systems”

Kurt Gödel 英訳 by Bernard Meltzer

https://goo.gl/PMMqYr

ヒルベルト・プログラムと「有限の立場」

1920年、ヒルベルトは、全ての数学に確実な基礎を提供するという壮大な計画を提出する。その主要な内容は、

- 数学は、形式言語で、明確なルールに従って記述されること。
- 数学的に正しい命題は、全て、この形式的システムの中で証明されること。
- この数学の形式化が矛盾を含まないことが証明されること。
  その無矛盾性の証明は、有限の数学的対象に対する「有限の立場」からの証明であるべきこと。
- 「実在的な対象」について、「理想的な対象」（例えば、「非可算集合」のような）を用いて行われた証明は、それらを用いなくても証明されるべきこと。
- 全ての数学的命題の真または偽を決定する方法が存在すること。

1930年　ゲーデルの不完全性定理 (1)

ゲーデルの不完全性定理とは、次のような定理である。

「ある形式的体系Sを作ったとしよう。その体系では、初等数論を定義でき、矛盾がないとする。その時、Sに属する命題Gで、Sの中では証明も反証もできないような命題Gが存在する。」

例えば、次のような命題Gを考えてみよう。

G:「Gは証明できない。」

この命題Gが証明できたとする。それは、Gは証明できないことを証明したことになる。

この命題Gの否定が証明できたとする。それはGは証明できること、すなわち、Gは証明できないことを証明したことになる。

嘘つきのパラドックス

これは、つぎのような「嘘つきのパラドックス」と、同じような構造である。

あるクレタ人が語った。「クレタ人の言うことは全て嘘である。」

このクレタ人の言うことが嘘ではなく正しいとしよう。それは、「クレタ人の言うことは全て嘘である。」というのが正しいと言うことなので、このクレタ人の言葉も嘘になる。

このクレタ人の言うことが嘘だとしよう。それは、彼が嘘をついていないことを意味する。

1930年　ゲーデルの不完全性定理 (2)

それだけだと、そんなものかと思われるかもしれないが、ゲーデルの不完全性定理が、数学の世界に強い衝撃を与えたのは、不完全性定理の次の形である。

Fを、初等数論を含む、無矛盾の形式的体系としよう。この時、この形式的体系Fの無矛盾性は、この体系Fの中では証明出来ない。

この結果は、ヒルベルト・プログラムが、実行不可能であることを意味していた。

Church-Turing Thesis

不完全性原理は、我々の認識の限界？

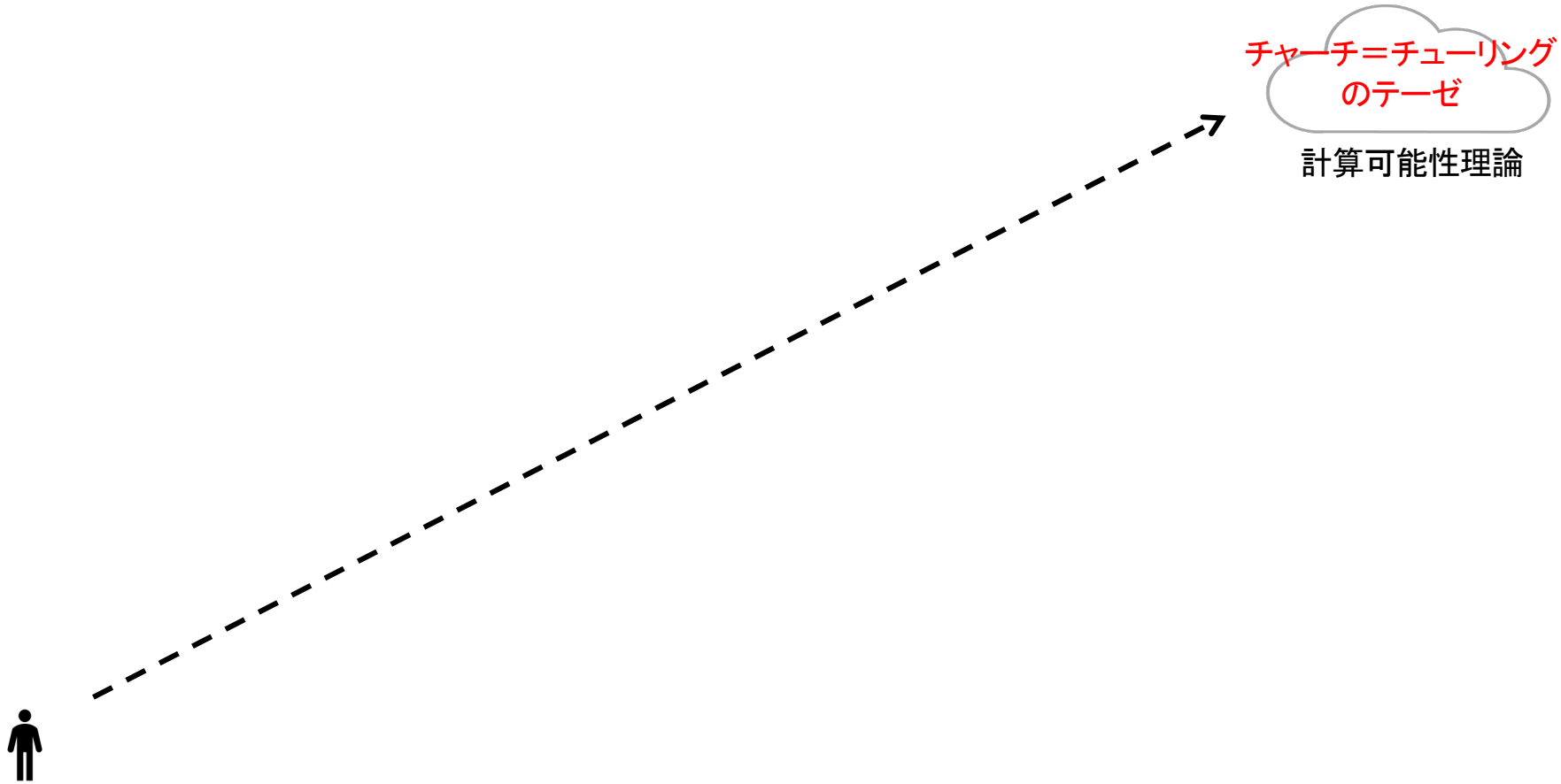
ただ、この不完全性定理は、確かに驚くべきものではあるのだが、ほとんど役に立たない定理である。

というのは、言い過ぎなのだが、この定理から、我々人間の（同時に、それは人工知能にとっての）認識能力の限界を議論するのは、あまりに、荒い議論であると僕は考えている。

一つには、不完全性定理による原理的な「限界」以外にも、形式的演繹による認識には、実質的には、様々な「限界」が存在しているからだ。我々の形式的認識の「限界」を構成しているのは、不完全性定理だけではないのだ。

（あとで見るように、複雑性理論は、もっと精緻に、その「限界」を境界づけようとしている。）

50年代



「証明可能＝計算可能」なものには限界があることは、
1950年代には、一般に認められるようになった。
ただ、それは、我々の認識の「限界」としては、非常に
荒い、かつ、抽象的で原理的な限界を与えるものでしかなかった。

「計算とはなにか？」「証明とは何か？」
より基本的な「問い」の始まり

あと一つには、ゲーデルの不完全性定理は、「証明不可能性」「形式的体系の無矛盾性」についてのショッキングな定理なのだが、いかなる計算、いかなる証明が「可能」であるかについては、あまり、多くのことを我々に教えてはくれないのだ。

歴史的に見ると、不完全性定理は、証明あるいは計算の性質についての探求の中で生まれた「最後の結論」ではないのだ。事実は、その逆である。1930年代に、この分野は、不完全性定理に強烈な刺激を受けて、「証明とは何か？」「計算とは何か？」という基本的な「問い」に突き進むことで発展する。

知的な展開としては、それに少し先行した、ハイゼンベルグやシュレジンガーやディラックといった若い天才たちが大活躍した量子力学の基礎付けの時代と比肩しうる、エキサイティングな「知的な黄金時代」だったと思う。（現代は、本当に、「イノベーティブ」なのかと、ふと思う。）

様々な「計算可能性」へのアプローチと その同値性の認識

ゲーデルの結果を受けて、「計算可能性」についての探求が一斉に始まる。

- ゲーデル＝エルブランの「帰納関数論」
- チャーチの「ラムダ・カリキュラス」
- チューリング＝ポストの「チューリング・マシン」

が続々と登場する。

この時代の白眉は、「計算可能性」についての見かけは全く異なるこれらのアプローチが（もちろん、それは、ゲーデルの結果を再現できるものだ）、次々と、「同値」であることが証明されていくことだ！

チューリング・マシンの「停止問題」

関数halts(f)は、ある関数fの計算が停止するなら真を、そうでないなら偽を返すものとする。

この時、次の関数gを考える。

```
def g():
    if halts(g):
        loop_forever()
```

halts(g)が真を返すなら、それは、関数gの計算が停止すると言うことなのだが、gの定義を見れば、その時関数gの計算は、無限ループに入って終わらない。

halts(g)が偽を返すなら、それは関数gの計算が停止しないと言うことなのだが、gの定義を見れば、それはすぐに終わる。

チャーチ＝チューリングのテーゼ

1940年代には、今日、「チャーチ＝チューリングのテーゼ」と呼ばれる「計算可能性」についての認識は、確立されることになる。（これが、「複雑性理論」の第一世代である。）

「チャーチ＝チューリングのテーゼ」というのは、次のような提案である。

Every effectively calculable function (effectively decidable predicate) is general recursive

すべての実効的に計算可能な関数（実効的に決定可能な述語）は、一般帰納的である。

計算主義の源流
TuringとChomsky

チューリングの問題提起
「機械は考えることができるか？」

チューリングの問題提起
「機械は考えることができるか？」

ゲーデルの不完全性定理は、「人間の認識の限界」について、山のような議論を巻き起こした。これらの有象無象の議論の中で、出色のものが一つあった。それは「機械は考えることができるか？」というチューリングの問題提起だったと僕は考えている。

数学的な命題であるゲーデルの定理を、いろいろ拡大解釈して人間の認識の限界と結びつけるのではなく、「機械だって、人間のように考えることができるんじゃないの？」と、機械の認識の可能性を対置したのだから。

今から約70年前、「機械は考えることができるか？」と問いかける論文がでる。今日の人工知能研究の出発点と言っていいチューリングの論文である。

この論文には、とても印象的な一節がある。

「「機械は考える事が出来るか？」という、最初に掲げた間題が、今では議論にも値しない程無意味なものである事は私も認めよう。… しかし、それにもかかわらず、今世紀の終わりには人々の意見が大きく変化して、ついには、矛盾していると考えることなく『機械の恩考』について語ることができるようになるであろうと私は信じている。」

チューリングの予想は、70年の時を超えて、見事に的中した。今では、「人工知能」「AI」という言葉を、メディアで聴かない日はないと言っていいくらいである。「人工知能」の可能性を信じるものの一人として、僕は、こうした変化を歓迎している。

ただ、僕は、皆が一斉に、「矛盾していると考えることなく『機械の恩考』について語」っていることに、少し、疑問を持ち始めている。

チューリングの予想は的中したのだが、「人々の意見が大きく変化する」その仕方は、かつてのチューリングの先見的で勇気ある深い問題提起に応えるものではないと感じている。メディアをみる限り、残念ながら、それは表面的で、ある場合には科学的でさえない。

楽観論

1957年、サイモンは次の様な予言を行った。

「こうした、いくつかの発展と、この分野での研究が進みつつあるスピードを基礎にして、きたる十年以内に以下の事が実現すると私はすすんで予言しようと思う。

1. 10年以内に、コンピューターは、チェスの世界チャンビオンとなるであろう。
2. 10年以内に、コンピューターは、重要な新しい数学の定理を発見し、証明するであろう。
3. 十年以内に、コンピューターは、批評家たちにかなりの美的な価値を持つと認められるであろう音楽を作曲するであろう。
4. 10年以内に、心理学のほとんどの理論は、コンピューター・プログラムの型をとるか、あるいはコンピューター・プログラムの特徴についての量的な言明という型をとるであろう。

悲観論

こうした楽観論の破産が明確になった70年代に入ると、こんどは逆に、専門家の中では、チューリングの見解をも否定的にとらえる傾向が有力になった様に恩われる。石田は「人工知能研究の批判」と題する論文で次の様に語る。(1972年)

「しかし六十年代を通しての研究経過を見てみると、計算機は、人間の頭脳労働のある局面を代行することはできるものの、結局”考えることは以来ない機械”であり、将来ともに”考える機械”の実現はほとんど絶望的なことがはっきりしたと恩われるのである」。「さて、常識的な意味で、計算機が”考える機械”でないことははっきりした。これは人類にとって祝福すぺきことであろう。外国語の翻訳をしたり、話しことぱを聞きわけたりする機械の出現もほとんど絶望的である。」

原理的可能性と具体的可能性

しかしながら、我々は、この二つの予言に大きな違いがあると考えるものである。それは、チューリングが「機械は考えることができるか」という、人工知能の原理的可能性を論じているのに対して、サイモンは、そうした問題を論じる事はなく、直接にその具体的可能性を予言している事である。こうした視点から言うなら、先にみた石田の議論は、現在の技術的な到達点から、人工知能の原理的な不可能性を論じたものだということができる。我々は、こうした立論にも同意しない。二つの可能性は、たしかに密接に結ぴついてはいるが、概念的には明確に区別されねばならないし、その区別は、方法論的にも重要な意味を持つ様におもわれる。

文法の階層性
Chomsky Hierarchyについて

Chomsky Hierarchy

- 形式的言語の形式的文法は、次のような階層をなすことが知られている。これをChomsky Hierarchyと呼ぶ。
  - タイプ-0 文法は、全ての形式文法を包含する。
  - タイプ-1 文法は、文脈依存言語を生成する。
  - タイプ-2 文法は、文脈自由言語を生成する。
  - タイプ-3 文法は、正規言語を生成する。
- 機械が、この階層の中に位置付けられるどの形式的な文法を理解・学習できたかを考えることができる。

https://goo.gl/lR960c

Chomsky Hierarchy

Grammar	Languages	Automaton	Production rules (constraints)
Type-0	Recursively enumerable	Turing machine	$\alpha \to \beta$ (no restrictions)
Type-1	Context-sensitive	Linear-bounded non-deterministic Turing machine	$\alpha A \beta \to \alpha \gamma \beta$
Type-2	Context-free	Non-deterministic pushdown automaton	$A \to \gamma$
Type-3	Regular	Finite state automaton	$A \to a$ and $A \to aB$

https://en.wikipedia.org/wiki/Chomsky_hierarchy

Chomsky Hierarchyと Deep Learningでの文法理解の取り組み

- 機械が、この階層の中に位置付けられるどの形式的な文法を理解・学習できたかを考えることができる。
- Hochreiterの仕事は、機械が、有限オートマトンで表現される正規文法を、理解・学習できることを示唆している。
- Karpathyの仕事は、機械が、BNF(Backus-Naur Form)で表現されるような、Latex, プログラム言語の文脈自由文法を、理解・学習できることを、強く示唆している。

Chomsky Hierarchyと
Deep Learningでの文法理解の取り組み

	文法	例	Deep Learning
Type-0	帰納的可算	チューリングマシン	
Type-1	文脈依存文法	自然言語*	?
Type-2	文脈自由文法	プログラム言語	Karpathy **2015年**
Type-3	正規文法	有限オートマトン	Hochreiter **1997年**

自然言語の多くの文法は、Contex-Freeで記述できる。
Mildly Context Sensitive Languages',proposed by Aravind Joshi.

Chomsky Hierarchyと Deep Learningでの文法理解の取り組み

	文法	例	Deep Learning
Type-0	帰納的可算	チューリングマシン	
Type-1	文脈依存文法	自然言語	Google Neural Machine Translation System ？ **2016年**
Type-2	文脈自由文法	プログラム言語	Karpathy **2015年**
Type-3	正規文法	有限オートマトン	Hochreiter **1997年**

自然言語の文法は、この階層のどこに位置付けられるか？

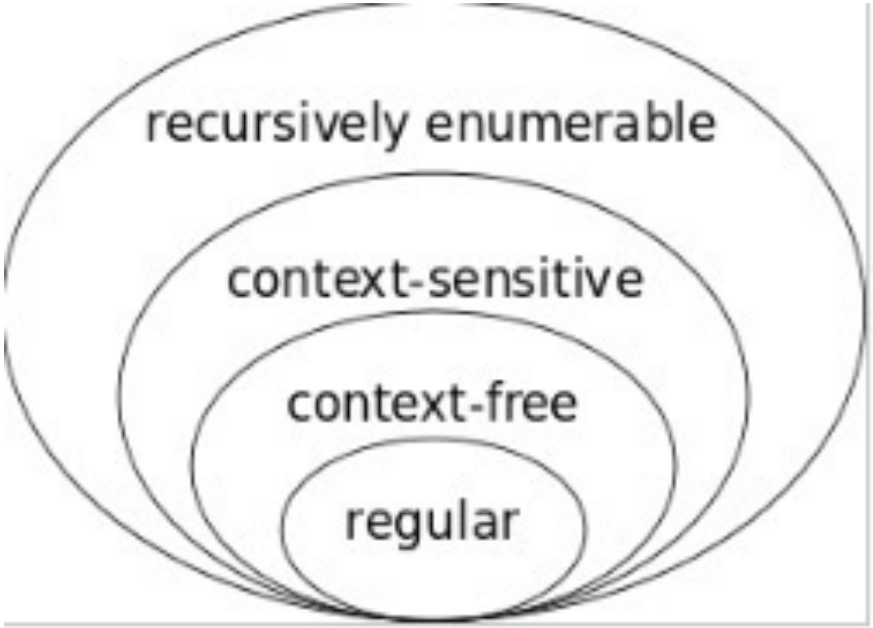
- 問題は、自然言語の文法が、この階層のどこに位置付けられるかということである。

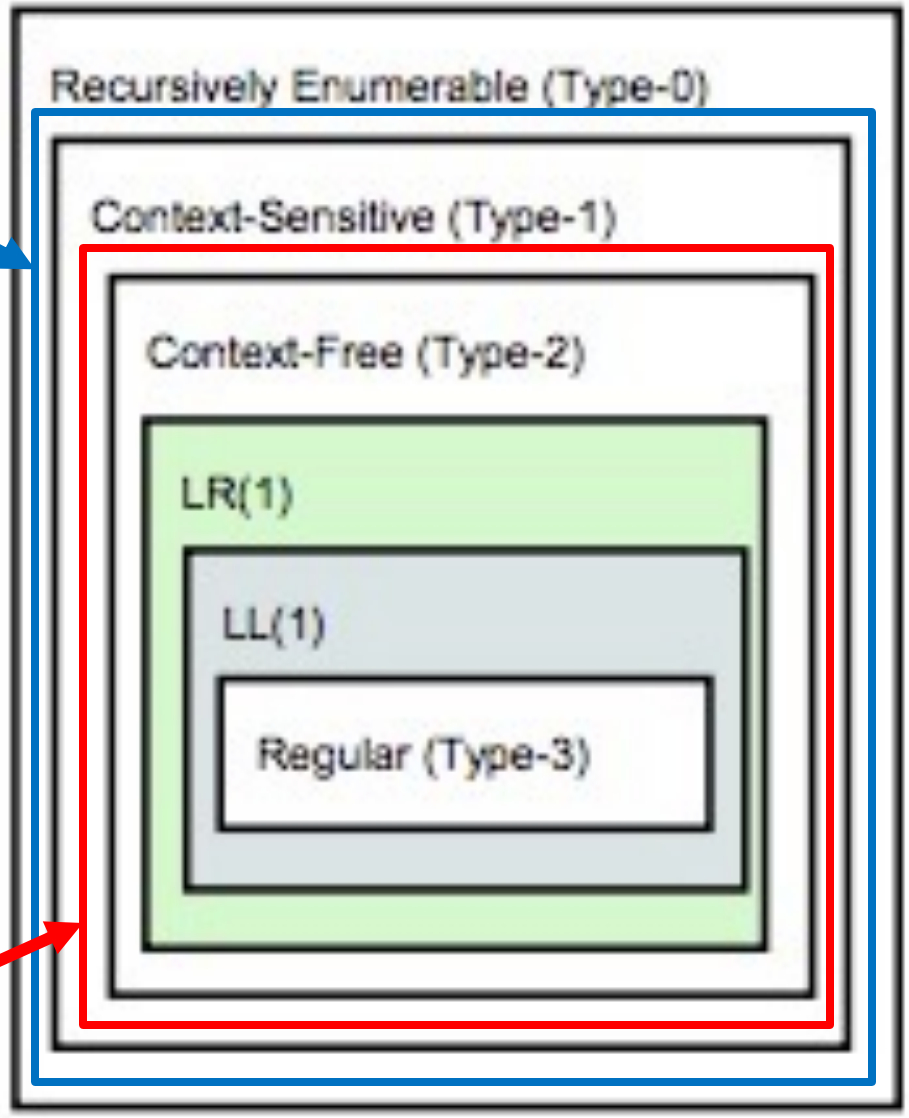
- 元の4階層の区分は、基本的だが粗いものである。Context-Freeな文法の中にも、LRやLLがあるように。
- かつてのChomskyは、英語は Regular 文法ではないと明確に述べたが、それが Context-Freeであるかどうかは、明確には判断を下していないようにも見える。ただ、自然言語は、Context-Free性を満たさないというのが、大方の意見のように見える。それは、Context-Freeを超えた、Mildly Context-Sensitive な文法だという。（次の図の、赤い四角の部分）

Minimalist ProgramとMerge

- 現在のChomskyのMinimalist Programでは、自然言語の文法を、文法の階層の中に位置付けようとする問題意識は、あまりないようにも見える。
- Category文法の人たちは、Minimalist文法が、weakly equivalent to multiply context-free ones だと言っているようだ。
- ただ、Minimalist Programの中心的概念であるMergeについては、それがRecursiveであることが強調されている。Recursiveというのは、Recursive Enumerableの一部分で、決定可能なものである。（図の青い四角の部分）
- 人間の言語能力と人間の数学的能力の違いを考えるとき、RecursiveとRecursive Enumerableの違いは、大きな意味を持つ。

Recursive Language
“Merge” is recursive





自然言語？

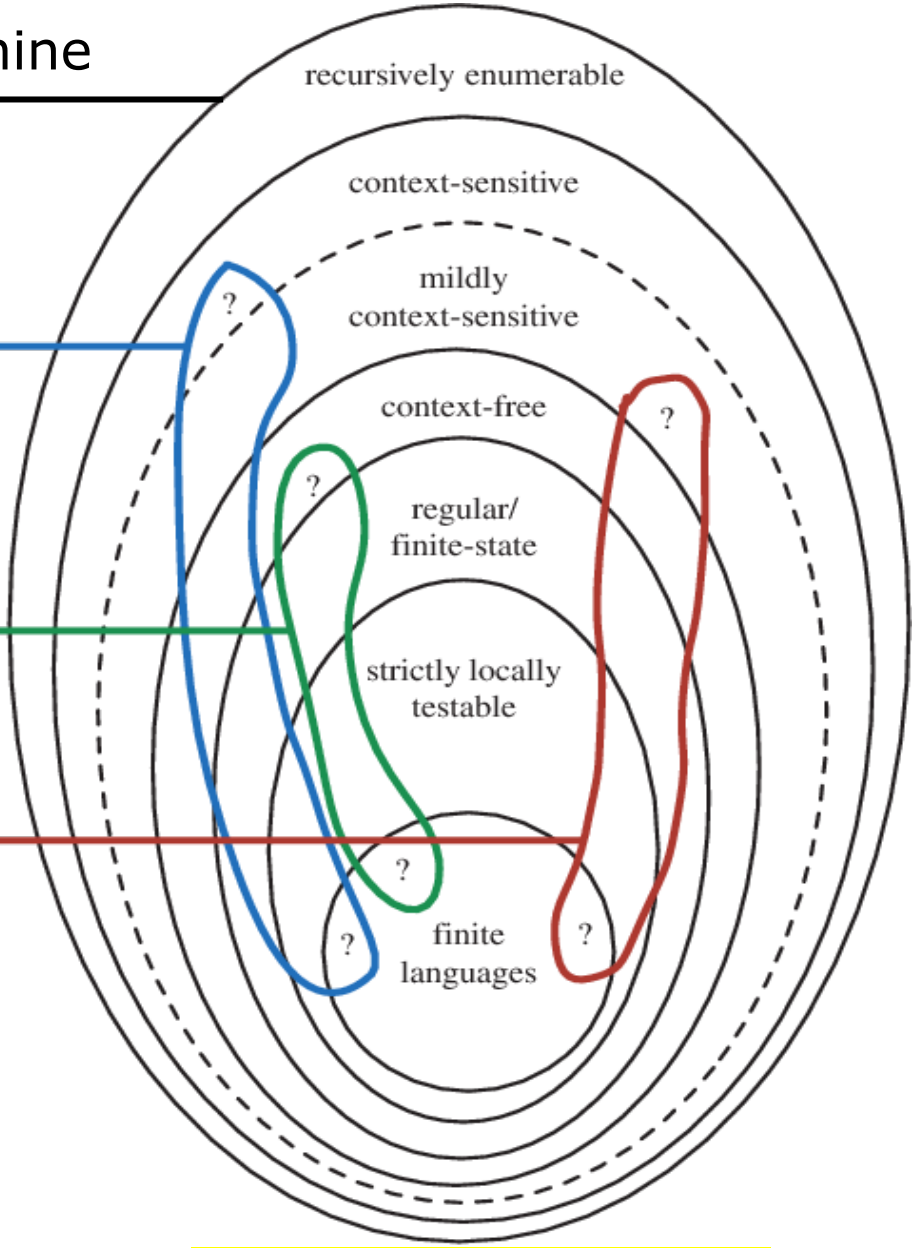
Mildly Context Sensitive Languages

Turing Machine

言語

鳥のさえずり

音楽



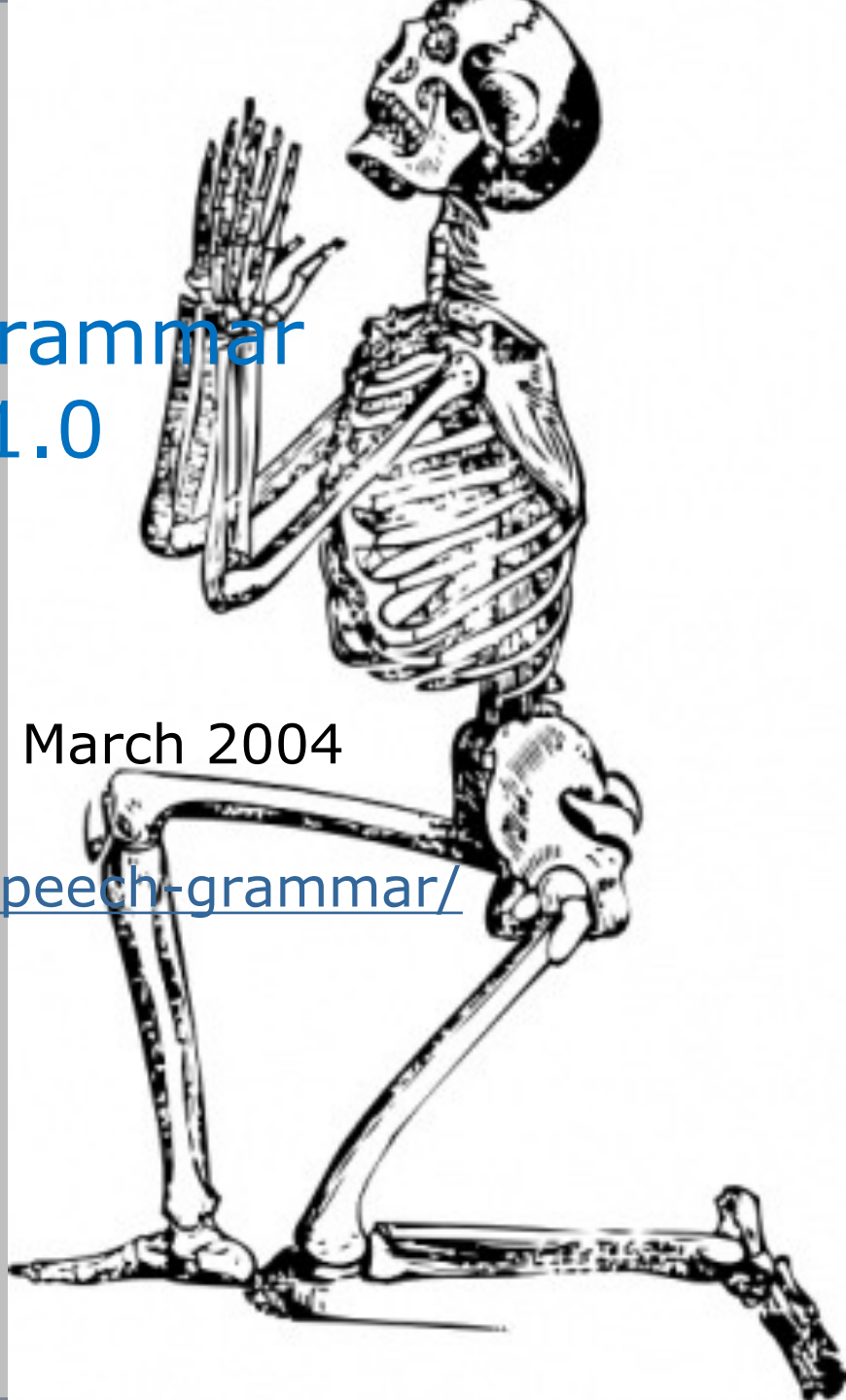
https://goo.gl/jmecp3

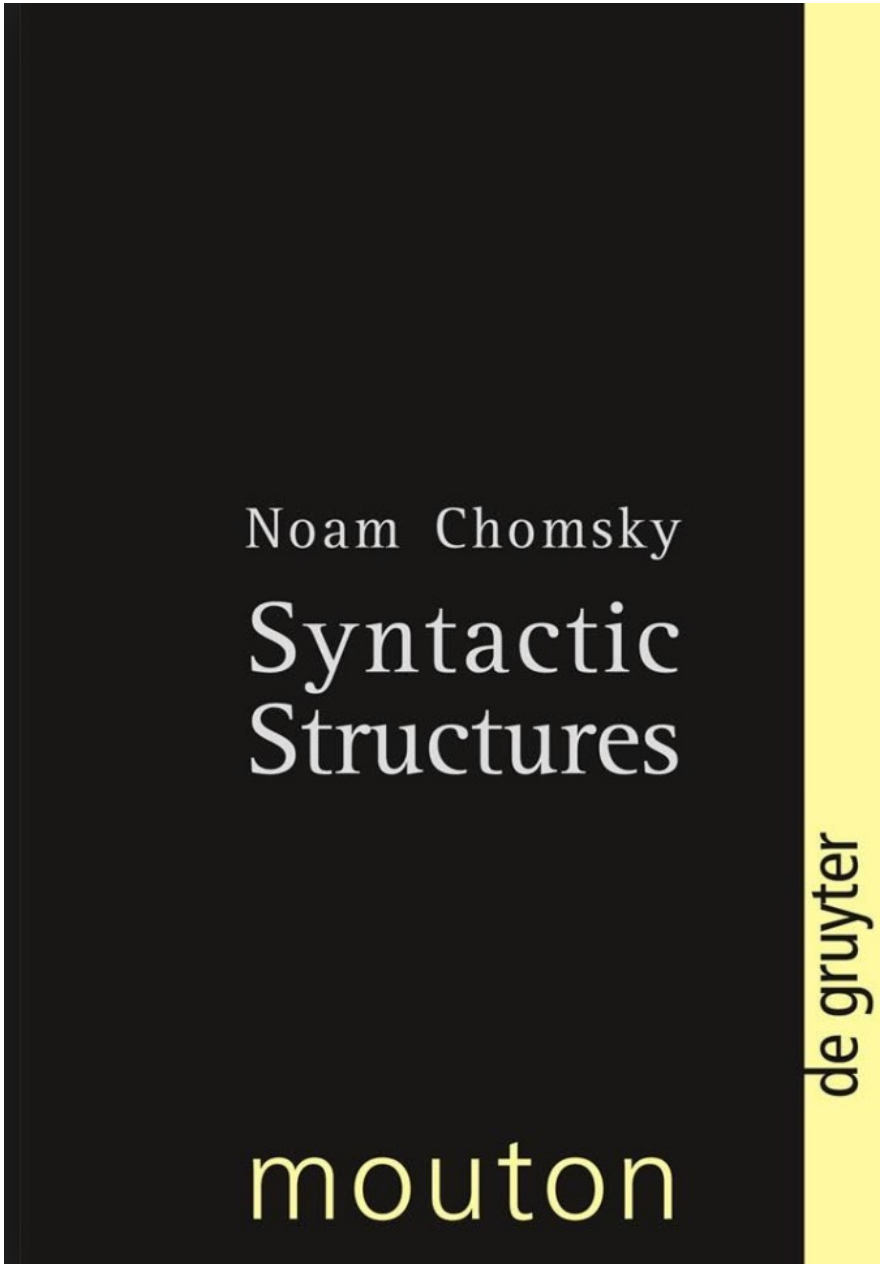
Speech Recognition Grammar Specification Version 1.0

W3C Recommendation 16 March 2004

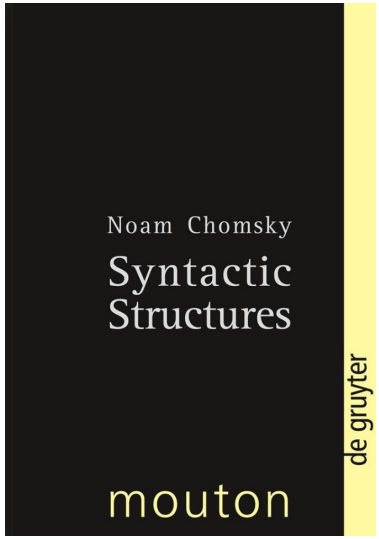
https://www.w3.org/TR/speech-grammar/

ここでの「文法」は、正規文法でしかない。
それでは、コンピューター言語もコンパイラーも、作れない。




Noam Chomsky
Syntactic Structures
mouton
de gruyter

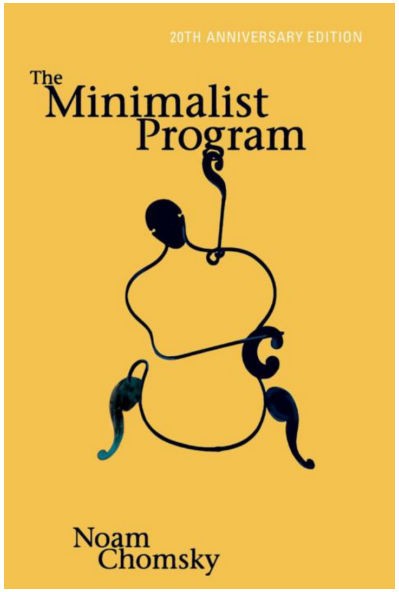
1957年



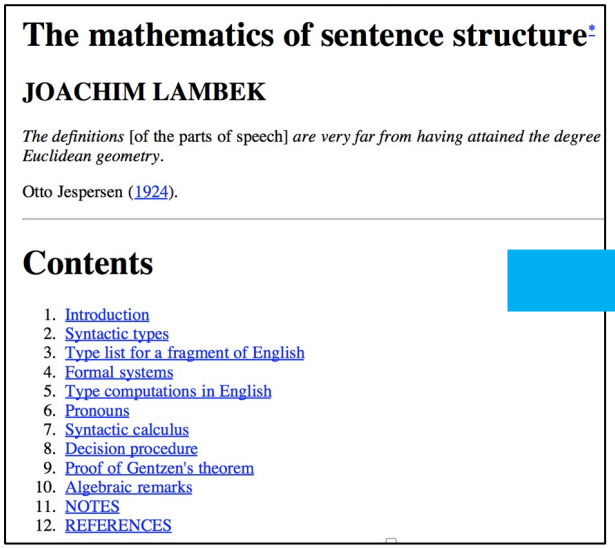
Chomsky

48年

1995年



1958年



The mathematics of sentence structure*

JOACHIM LAMBEK

The definitions [of the parts of speech] *are very far from having attained the degree Euclidean geometry.*

Otto Jespersen (1924).

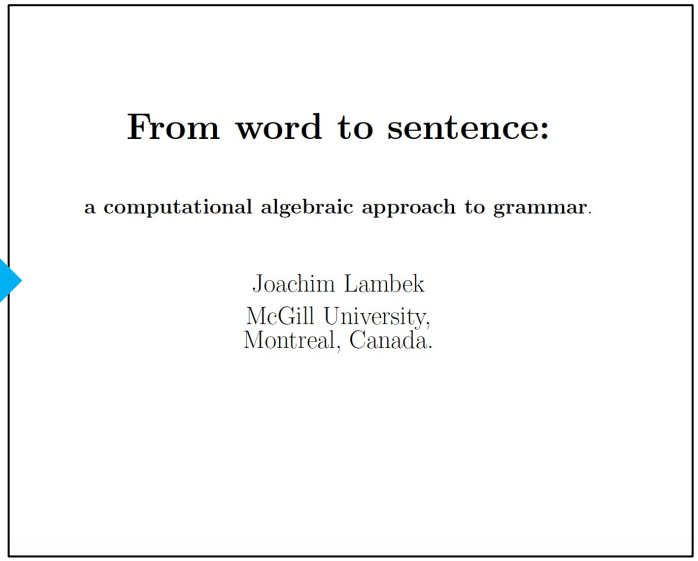
Contents

1. Introduction
2. Syntactic types
3. Type list for a fragment of English
4. Formal systems
5. Type computations in English
6. Pronouns
7. Syntactic calculus
8. Decision procedure
9. Proof of Gentzen's theorem
10. Algebraic remarks
11. NOTES
12. REFERENCES

Lambek

50年！

2008年



From word to sentence:

a computational algebraic approach to grammar.

Joachim Lambek
McGill University,
Montreal, Canada.

「有限の立場」と「実効的計算可能性」

「有限の立場」と証明論

The finitary point of view

As a condition for the use of logical inferences and the performance of logical operations, something must already be given to our faculty of representation, certain extra-logical concrete objects that are intuitively present as immediate experience prior to all thought. If logical inference is to be reliable, it must be possible to survey these objects completely in all their parts, and the fact that they occur, that they differ from one another, and that they follow each other, or are concatenated, is immediately given intuitively, together with the objects, as something that can neither be reduced to anything else nor requires reduction. This is the basic philosophical position that I consider requisite for mathematics and, in general, for all scientific thinking, understanding, and communication.

Hilbert, 1926

Proof Theory

The goal of Hilbert's program -- to give consistency proofs for analysis and set theory within finitist mathematics -- is unattainable, the program is dead.

The mathematical instrument, however, that Hilbert invented for attaining his programmatic aim is remarkably well: *proof theory* has obtained important results and pursues fascinating logical questions . . .

we may ask ourselves whether the results of proof theory are significant for the foundational concerns that motivated Hilbert's program and, more generally, for a reflective examination of the nature of mathematics.

Wilfried Sieg, *Hilbert's Programs and Beyond*,

ゲンツェンの数論の無矛盾性証明

- 1936年、ゲンツェンは、ϵ_0までの超限帰納法を利用して、ペアノ数論の無矛盾性を証明した。
- ゲーデルの不完全性定理で、数論の無矛盾性証明は不可能だと思っている人が多いのだが、それはその体系の内部ではできないということ。
- ϵ_0までの超限帰納法は、ペアノ数論の内部には含まれていない。しかし、それは、「自然」な想定である。
- 残園なことに、ゲンツェンは、若くして夭折している。



Gerhard Karl Erich Gentzen、
1909年11月24日 - 1945年8月4日

https://goo.gl/6s9K1R

カテゴリー文法でのゲンツェンのシーケンスの利用

show me	flights	to	Boston
S/N	N	PP/NP	NP
$\lambda f.f$	$\lambda x.flight(x)$	$\lambda y.\lambda x.to(x,y)$	$BOSTON$
		PP (>)	
		$\lambda x.to(x, BOSTON)$	
		$N\backslash N$ (<)	
		$\lambda f.\lambda x.f(x) \wedge to(x, BOSTON)$	
	N (<)		
	$\lambda x.flight(x) \wedge to(x, BOSTON)$		
S (>)			
$\lambda x.flight(x) \wedge to(x, BOSTON)$			

行の上は、Syntax
行の下は、Semantics

SyntaxとSemanticsを同時に扱うことができる

https://goo.gl/PZE9oA

一般帰納関数の計算可能性
アッカーマン関数

一般帰納関数の計算可能性
アッカーマン関数

次のような関数を考える。

$$\mathrm{Ack}(m,n) = \begin{cases} n+1, & \text{if } m = 0 \\ \mathrm{Ack}(m-1,1), & \text{if } n = 0 \\ \mathrm{Ack}(m-1,\mathrm{Ack}(m,n-1)), & \text{otherwise} \end{cases}$$

この関数は、一般帰納関数として定義されているので、チャーチ＝チューリングのテーゼの条件を満たしている。

ただ、m,nの値が、少しでも大きくなると、この関数の値は、爆発的に増大して、普通のコンピュータでは、事実上、計算が不可能になる。

次ページに、Ack(m,n)の値を示す。

A(*m*, *n*) の値

*m**n*	0	1	2	3	4	n
0	1	2	3	4	5	$n+1$
1	2	3	4	5	6	$n+2 = 2+(n+3)-3$
2	3	5	7	9	11	$2n+3 = 2\times(n+3)-3$
3	5	13	29	61	125	$2^{n+3}-3$
4	13	65533	$2^{65536}-3$	$2^{2^{65536}}-3$	A(3, A(4, 3)) = $2^{2^{2^{65536}}}-3$	$\underbrace{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}}_{n+3\text{ twos}}-3$
5	65533	$\underbrace{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}}_{65536\text{ twos}}-3$	A(4, A(5, 1))	A(4, A(5, 2))	A(4, A(5, 3))	$\mathrm{A}(4,\mathrm{A}(5,n-1))$
6	A(5, 1)	A(5, A(6, 0))	A(5, A(6, 1))	A(5, *A*(6, 2))	A(5, A(6, 3))	$\mathrm{A}(5,\mathrm{A}(6,n-1))$

Cantorの順序数

「順序数」は、「自然数の順番に数える」ことをベースにした数の拡張である。もちろん、ベースになった自然数は、順序数である。この拡張は、無限に数え続けることを可能にすることを目的にしていると考えることができる。

ここでは、John Baezの ”Large Countable Ordinals” に多くを負っている。 https://goo.gl/h5HhrV

順序数

- ある順序数を、それより小さい全ての順序数の集合としてあらわす。

0={}　　ϕ

1={0}　　0

2={0, 1}　　0 < 1

3={0, 1, 2}　　0 < 1 < 2

4={0, 1, 2, 3}　0 < 1 < 2 < 3

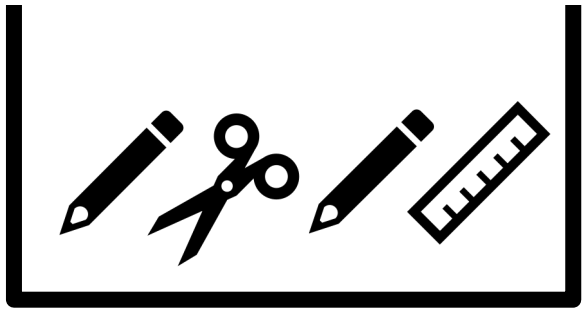
…

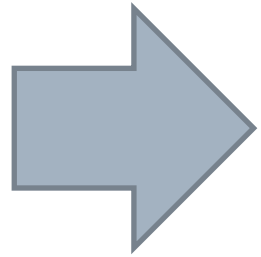
n={0, 1, 2, … , n-1}　0 < 1 < 2 < … < n-2 < n-1

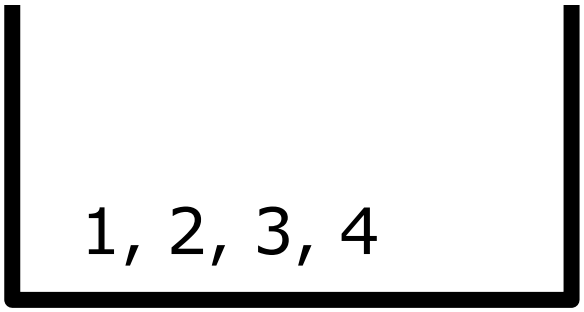
有限の順序数

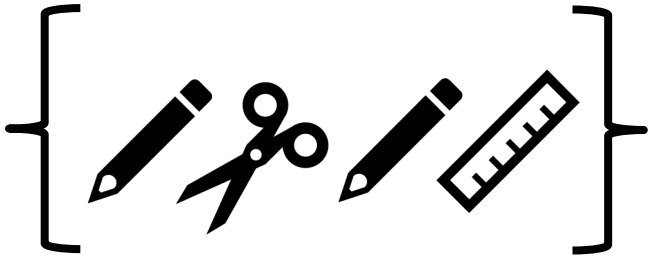
- 順序数が有限の時、それは、ある自然数に一致する。
  それはまた、その集合の要素の数（基数）とも一致する。
- 有限の時、
  順序数 ＝ 基数 ＝ 順序数の要素の数 である

順序数の系列：0, 1, 2, 3, 4, 5, 6, 7, 8, 9, …
基数の系列：　0, 1, 2, 3, 4, 5, 6, 7, 8, 9, …





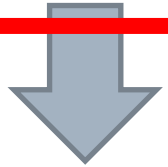






数える

{ 1, 2, 3, 4 } 順序数
Ordinals



「ある順序数を、それより小さい全ての順序数の集合としてあらわす。」という定義は、0から始まっているが、先に「数えるを考える」でみたこの図での説明は、1から始まっている。
そうした違いはあるが、両者は同じものである。

4 個

個数

基数
Cardinals

ω：有限な順序数全ての集合

- 全ての有限な順序数全ての集合 {0, 1, 2, 3, … }を考える。この集合を ω (omega)で表す。ωは全ての自然数からからなる無限集合である。

  $\omega = \{0, 1, 2, 3, \dots\}$

- 0,1,2,3,…< ωである（どの自然数nもωより小さい）と考えれば、このωの定義は、「ある順序数を、それより小さい全ての順序数の集合としてあらわす。」という順序数の定義に当てはまるので、ωも順序数である。

- 今度は、{0, 1, 2, 3, … , ω}を考える。0,1,2,3,…< ωであるとすれば、この集合もある順序数に対応する。
  この順序数を ω+1 で表す。

  $\omega+1 = \{0, 1, 2, 3, \dots, \omega\}$

ちょっと、ここで立ち止まろう。

- ω= {0, 1, 2, 3, … }は、無限の要素を持っている。

  この集合に終わりの要素はない。

- また、ω+1 = {0, 1, 2, 3, … , ω}の右辺のオメガには、直前の要素はない。

- ただし、0,1,2,3,…< ω とすれば、ω+1の要素すべてにきちんと順序が定義されている（整列されている）。だから、ω＋1もまた、順序数である。

後継順序数と極限順序数／可算順序数

- ある順序数αについて、$\beta=\alpha+1$の形で表される順序数βを、後継順序数という。(「孤立点」)
  - 自然数 1, 2, 3 等々、有限な順序数は、後継順序数である。
  - $\omega+1$は、無限な順序数だが、後継順序数である。
  - $\omega+2$は、無限な順序数だが、$(\omega+1)+1$で後継順序数である。
- ωは、直前の要素を持たず、$\beta=\alpha+1$の形では表されない。こうした順序数を、極限順序数という。(「集積点」)

- 順序数は、「数える」という操作と結びついている。それは、「順番に数える」という操作の無限への拡大である。この章で扱う順序数は、様々な無限の型をとるが、基本的にはすべて「可算(Countable)」である。

順序数の和

- 順序数αと順序数βの和α+βは、αの後ろにβをつなげて、つなぎめのところで新しい順序関係を導入して、前後で一貫した順序が保たれるようにしたものと思えばいい。
- だから、ω+1は、{0,1,2,3, …,0}となる。同じ0が2度出てきているように見えるが、グリーンの0はαに属し、赤の0はβに属する。これを繋げて一つの順序にするには、0,1,2,3, … < 0 とすれば良い。これは、0がωだと思っていいことを示す。
- 逆に、1+ωは、{0,0,1,2,3,…}で、0<0 という関係を導入すればいいいのだが、この時、この順序関係は、{0,1,2,…}と同じものになる。よって1+ω=ωである。
- こうして、1+ω = ω ≠ ω+1 である。

順序数 ω＋ω

- この調子で、どんどん順序数を数えていこう。

$$\omega = \{0, 1, 2, 3, \dots\}$$

$$\omega + 1 = \{0, 1, 2, 3, \dots, \omega\}$$

$$\omega + 2 = \{0, 1, 2, 3, \dots, \omega, \omega + 1\}$$

$$\omega + 3, \omega + 4, \omega + 5, \dots$$

$$\omega + \omega = \{0, 1, 2, 3, \dots, \omega, \omega + 1, \omega + 2, \omega + 3, \dots\}$$



- ω＋ωは、後継順序数の並びでない箇所が二箇所ある。
  このω＋ωを、ω2と表す。ωが二つという意味である。

順序数の積

- 順序数αと順序数βの積αβは、αをβ個つなげたものと思えばいい。もちろん、つなぎめのところで新しい順序関係を導入して、前後で一貫した順序が保たれるようにする。
- 例）ω= {0, 1, 2, 3, … } だから
  ω2 = {0, 1, 2, 3, …, 0, 1, 2, 3, … }
  これを繋げて一つの順序にするには、0,1,2,3, … < 0 とすれば良い。これは、0がωだと思っていいことを示す。
  だから、ω2 = {0,1,2,3,…,ω,ω+1,ω+2,ω+3,…}
- 2ω = {0,1,0,1,0,1,0,1,…[0,1の繰り返しω回] }
  ここで、1<0, 1<0, 1<0, … という順序を導入すればいい。
  これは、結局、ω= {0, 1, 2, 3, … } と同じ順序にできる。
  だから、2ω = ω
- 2ω = ω ≠ ω2

順序数 ωω (= ω^2)

- どんどん順序数を数え続ける。

ω2 = {0,1,2,3,…,ω,ω+1,ω+2,ω+3,…}

$$\omega 2 + 1, \omega 2 + 2, \ldots$$

$\omega 3.$

$\omega 4.$

$\omega 5$

⋮

ωω

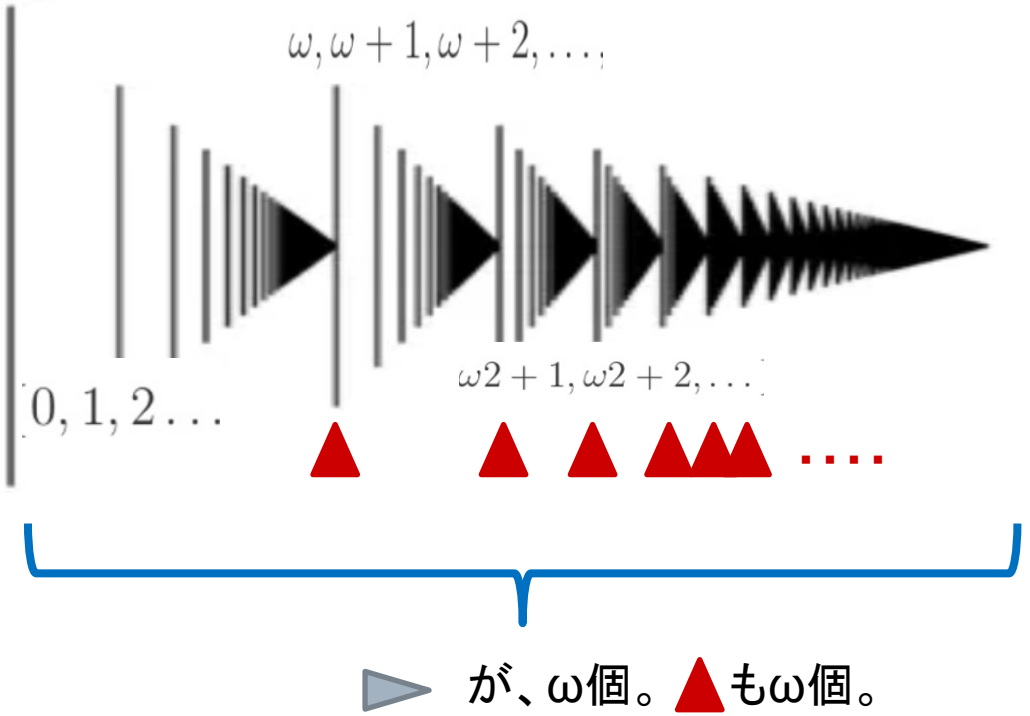
$$\omega^2 = \{0, 1, 2 \ldots \omega, \omega + 1, \omega + 2, \ldots, \omega 2, \omega 2 + 1, \omega 2 + 2, \ldots\}$$

ω^2 はどういう集合か？

- ω巻からなる百科事典を考える。各巻がωページからなるとすると、この百科事典は、ω^2ページを持つ。
- ただ、この集合はイメージできるかもしれないが、順序の様子を見るにはわかりにくい。大事なことは、全体で一つの順序をなしていることだから。
- そこでこうする。この本のページは、すべて一つ前のページの1/2の厚さを持ち、かつ、各巻は一つ前の巻の1/2の厚さを持つとする。本当はページ数・巻数が大きくなるたびに、無限の値は大きくなるのだが、それでは、限られた紙面では表示できなくなる。進めば進むほど小さくなるように、「見える化」するのだ。

ω^2 のイメージ

ω^2 =



超限順序数を「見える化」するサイトがある！

David Madore: https://goo.gl/uv1uWq

順序数 ω^ω

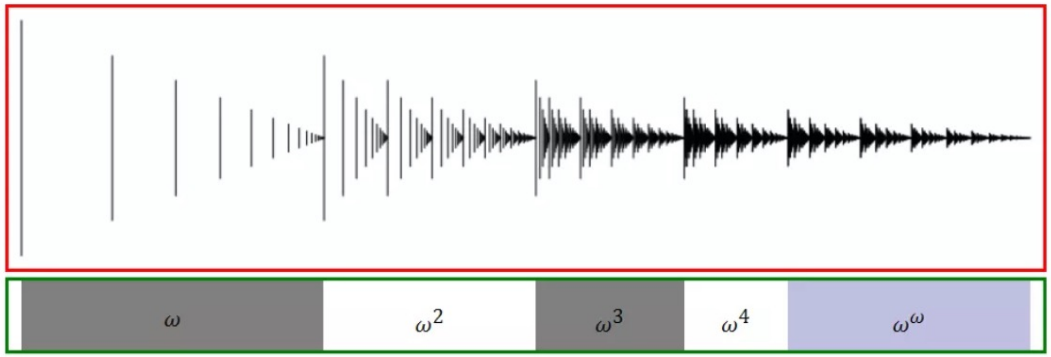
- まだまだ続く。

$$\omega^2 + 1, \omega^2 + 2, \ldots$$
$$\omega^2 + \omega, \omega^2 + \omega + 1, \omega^2 + \omega + 2, \ldots$$
$$\omega^2 + \omega^2 = \omega^2 2$$
$$\omega^2 3$$
$$\omega^2 4$$
$$\vdots$$
$$\omega^2 \omega = \omega^3$$
$$\vdots$$
$$\omega^\omega$$

順序数 ω^ω のイメージ

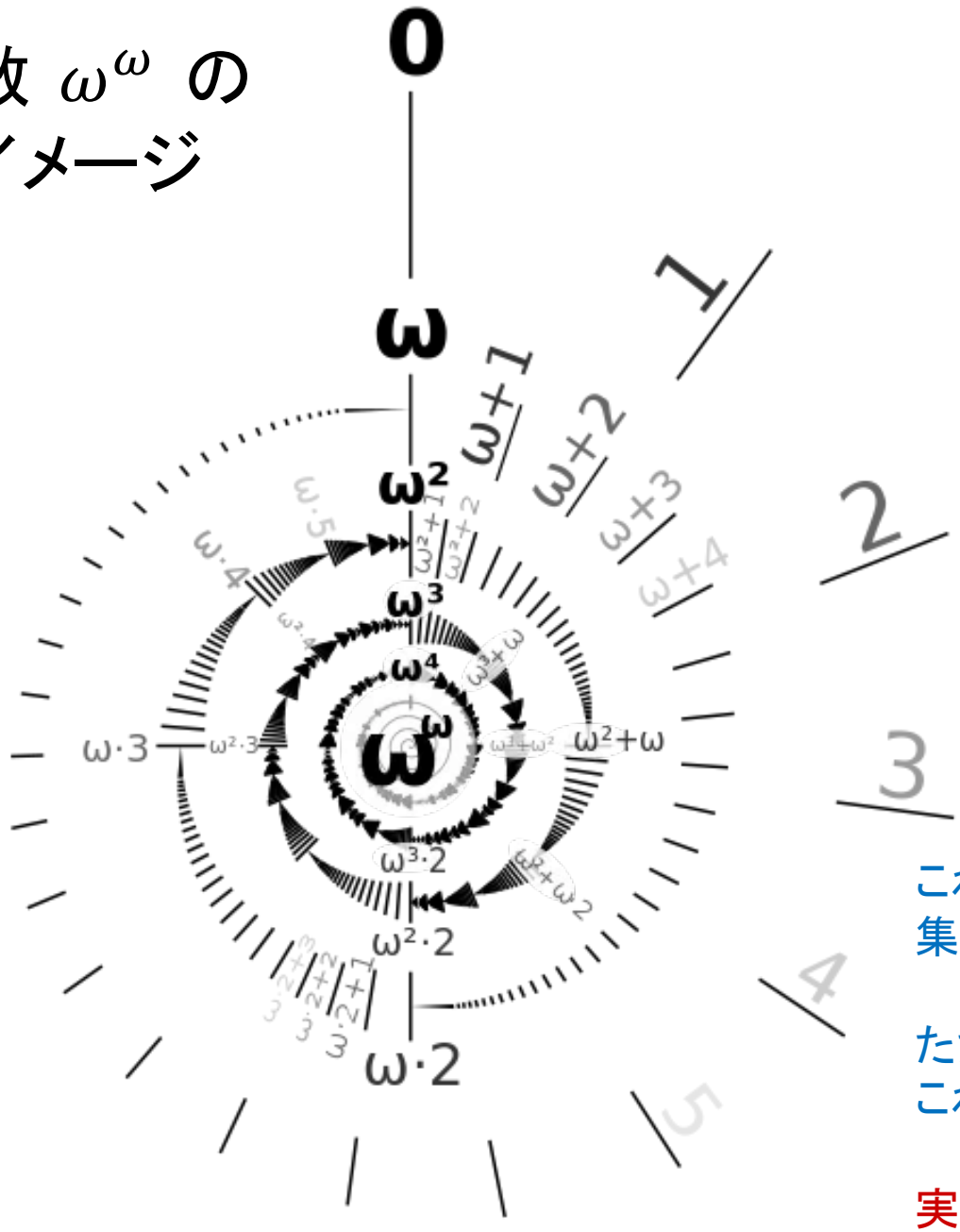
- ωページからなる本を考え、ω巻からなる百科事典を考え、ω種類の百科事典が入った入った本棚を考え、ω個の本棚が入ったフロアを考え、ω階のフロアからなる図書館を考え、ω個の図書館を持つ都市を考え、ω個の都市を持つ惑星を考え、ω個の惑星をもつ恒星系を考え、ω個の恒星系からなる銀河を考え、…こんなのがずっと続く。
- でも、このシステムで、ある本のあるーページを、具体的に指定しようと思ったら、有限個の自然数の指定で、このページに行き当たるから、この順序数は可算である！

ω^ω =




970
950
920
800
120

順序数 ω^ω の別のイメージ



これは、なかなか秀逸。
集積点が直線上に並ぶ。

ただし、ω^ω までなのだが。
これで終わりではない。

実は、まだまだ続く。

まだ、終わらない！

ω^ω

$\omega^\omega + 1$

$\omega^\omega 2, \ldots, \omega^\omega 3, \ldots, \omega^\omega 4, \ldots$

$\omega^\omega \omega = \omega^{\omega+1}$

$\omega^\omega \omega^2, \ldots, \omega^\omega \omega^3, \ldots, \omega^\omega \omega^4, \ldots$

$\omega^\omega \omega^\omega = \omega^{\omega+\omega} = \omega^{\omega 2}$

$\omega^{\omega 3}, \ldots \omega^{\omega 4}, \ldots \omega^{\omega 5}, \ldots$

$\omega^{\omega\omega} = \omega^{\omega^2}$

$\omega^{\omega^3}, \ldots \omega^{\omega^4}, \ldots \omega^{\omega^5}, \ldots$

ω^{ω^ω}

まだ、終わらない！

$$\omega^{\omega^\omega}$$

$$\omega^{\omega^\omega}+1, \omega^{\omega^\omega}+2, \ldots$$

$$\omega^{\omega^\omega}+\omega, \ldots, \omega^{\omega^\omega}+\omega 2, \ldots, \omega^{\omega^\omega}+\omega 3, \ldots$$

$$\omega^{\omega^\omega}+\omega^2, \ldots, \omega^{\omega^\omega}+\omega^3, \ldots$$

$$\omega^{\omega^\omega}+\omega^\omega$$

$$\omega^{\omega^\omega}+\omega^{\omega^\omega} = \omega^{\omega^\omega} 2$$

$$\omega^{\omega^\omega} 3, \ldots, \omega^{\omega^\omega} 4, \ldots, \omega^{\omega^\omega} 5, \ldots$$

$$\omega^{\omega^\omega}\omega = \omega^{\omega^\omega+1}$$

そろそろやめたい

$$\omega^{\omega^\omega+\omega}, \ldots, \omega^{\omega^\omega+\omega 2}, \ldots, \omega^{\omega^\omega+\omega 3}, \ldots$$

$$\omega^{\omega^\omega+\omega^2}, \ldots, \omega^{\omega^\omega+\omega^3}, \ldots, \omega^{\omega^\omega+\omega^4}, \ldots$$

$$\omega^{\omega^\omega+\omega^\omega} = \omega^{\omega^\omega 2}$$

$$\omega^{\omega^\omega 3}, \ldots, \omega^{\omega^\omega 4}, \ldots$$

$$\omega^{\omega^\omega\omega} = \omega^{\omega^{\omega+1}}$$

$$\omega^{\omega^{\omega+2}}, \ldots, \omega^{\omega^{\omega+3}}, \ldots, \omega^{\omega^{\omega+4}}, \ldots$$

$$\omega^{\omega^{\omega\omega}}, \ldots, \omega^{\omega^{\omega\omega\omega}}, \ldots, \omega^{\omega^{\omega\omega\omega\omega}}, \ldots$$

$$\epsilon_0 = \omega^{\epsilon_0}$$

べき乗についてのfixed point

$$\epsilon_0 = \omega^{\omega^{\omega^{\omega^{\omega^{\omega}}}}}$$

ω個

イプシロン・ゼロ

Cantor Normal Form

Theorem. Every ordinal α can be uniquely written as

$$\alpha = \omega^{\beta_1} c_1 + \omega^{\beta_2} c_2 + \cdots + \omega^{\beta_k} c_k$$

where k is a natural number, $c_1, c_2, \ldots, c_k$ are positive integers, and $\beta_1 > \beta_2 > \cdots > \beta_k \geq 0$ are ordinals.

$$\omega^{\omega^{\omega^{\omega^{1+1}+\omega+1}}} + \omega^{\omega^{\omega}+\omega^{\omega}} + \omega^{\omega+1+1} + \omega + 1$$

こんなのが、何か役に立つのであろうか？
そういう疑問を持った人が多いと思うが、
次のゲンツェンの仕事が、ひとつの回答になる。

Part III

計算複雑性理論

Agenda Part III
計算複雑性理論

- 計算複雑性理論
  - Gödel とNash
  - 計算複雑性理論とは何か
  - 計算複雑性理論の基本的クラス
- 複雑さについての多様なアプローチ
  - コルモゴロフの複雑性
  - チャイティンの不完全性定理
  - 我々に手の届かない有限について
Busy Beaver 問題
  - チャイティンとゲーデル

Gödel とNash

1950年代に入ると、時代の先鋭な知性たちは、今日の複雑性理論の原型とも言える認識に到達し始める。

スコット・アーロンソンは、複雑性理論の到達点を概観した論文 "N = NP?" https://goo.gl/TZUkgh の冒頭に、ジョン・ナッシュの 1955年のNSA（スノーデンがいた、あのSNAだ！）宛の手紙をあげている。かつては「機密文書」とされていて、いつかの時点で情報公開されたものらしい。

誰にも破れない暗号を作るためには、指数関数的な計算が必要な暗号を作ればいいことを述べている。また、そうすれば、それまでの職人技的な暗号破り（多分、チューリングやファインマンがしていたこと）は「過去」のものになると、明確に自覚している！

Nash

現時点での、私の一般的な予想は次のようなものです。:

ほとんどすべての十分に複雑なタイプの暗号化にとって、特に、鍵の異なる部分によって与えられた命令が、相互に複雑に相互作用して、それが暗号化の最終的な決定において影響を与えている場合、鍵の計算の平均的な長さは、鍵の長さ、すなわち、鍵の持つ情報の内容に関して、指数関数的に増加します。

もし、この予想が正しいと仮定すれば、この一般的な予想の重要性を理解するのは容易です。それは、実際的には誰も破れない暗号を設計することを、極めて簡単に実行できることを意味します。暗号が洗練されたものになるにつれて、熟練したチームなどによる暗号破りのゲームは、「過去」のものになっていくはずです。

この予想の性質は、たとえ特別な種類の暗号をとっても、私にはそれを証明できないようなものです。また、私は、それが証明されることも期待していません。」

---

https://www.nsa.gov/news-features/declassified-documents/nash-letters/assets/files/nash_letters1.pdf

現文は、こちら。

Now my general conjecture is as follows: For almost all sufficiently complex types of enciphering, especially where the ~~information~~ instructions given by different portions of the key interact complexly with each other in the determination of their ultimate effects on the ~~cipher~~ enciphering, the ~~number of~~ key computation length increases exponentially with the length of the key, or in other words, with the information content of the key.

The significance of this general conjecture, assuming its truth, is easy to see. It means that it is quite feasible to design ciphers that are effectively unbreakable. As ciphers become more sophisticated, the game of cipher breaking by skilled teams, etc., should become a thing of the past.

The nature of this conjecture is such that I cannot prove it, even for a special type of cipher. Nor do I expect it to be proven. But this

Gödel

僕は、ナッシュのNSAへの手紙は知らなかったのだが、有名なのは、1956年にゲーデルが、フォン・ノイマンに宛てた、次の手紙である。ゲーデルが、ほとんど完全に、問題の所在を把握していることがわかる。

我々が、一階の述語論理の全ての式Fに対して、全ての自然数について、長さn（長さ＝記号の数）のFの証明があるかどうかを決定することを可能にするチューリング機械を、簡単に構築することができるのは明らかです。

機械がこれを決定するのに必要とするステップの数をψ(F, n)とし、[特定のFについて] φ(n)＝maxψ(F, n)としましょう。問題は、最適なマシンで、φ(n)がどのくらい急速に大きなものになるかです。

我々は、[nとは独立な定数Kについて] φ(n) ≥ K･n　であることを示すことができます。[φ(n)は、nの一次式より大きいということ。]　実際に、実行時間が Knに比例する（またはKn^2に比例する）マシンが実際にあったとした場合 [「論理式Fに対して長さnのFの証明があるかどうか？」という問題が、nの一次式または二次式に比例する時間で解けるなら]　、これは最大級に重要な結果をもたらすことになります。

つまり、「決定問題」の解決不可能性にもかかわらず、YesかNoかの質問に答えるタイプの問題の場合、数学者の知的努力は完全に機械に置き換えることが可能であることを、明確に示すことができるでしょう。

我々が　実際に行うべきことは、単純に数nを選ぶだけです。そのnが非常に大きくても、もしも、マシンが [「長さnのFの証明があるかどうか？]という質問に] "No"という結果を出した場合には、さらに問題を考える理由はなくなります。

---

https://rjlipton.wordpress.com/the-gdel-letter/

もっとも、これらの50年代半ばの認識は、広く共有されたものではなかったかもしれない。（論文として公開されたものではなく「手紙」だった！）

計算複雑性理論とは何か

20世紀の初めにヒルベルトが挑戦したのは、全ての数学的問題の真偽を決定できる「純粋に機械的な手続き」をデザインすることだった。その夢は実現できなかったのだが、そうした過程が存在するのか、あるいは、存在しないとすれば何故そうなのかという疑問は残った。

そうした問いかけは、20世紀の二つの革命の登場を準備することになる。一つは、ゲーデル、チャーチ、チューリングらによる科学と哲学の分野での革命で、理性の限界自身を数学的分析の対象としたことだった。もう一つの革命は、テクノロジーの分野でのコンピュータの登場だった。それは、ヒルベルトの夢を実現したわけではなかったが、地球上の多くの人の日常生活を大きく変えた。

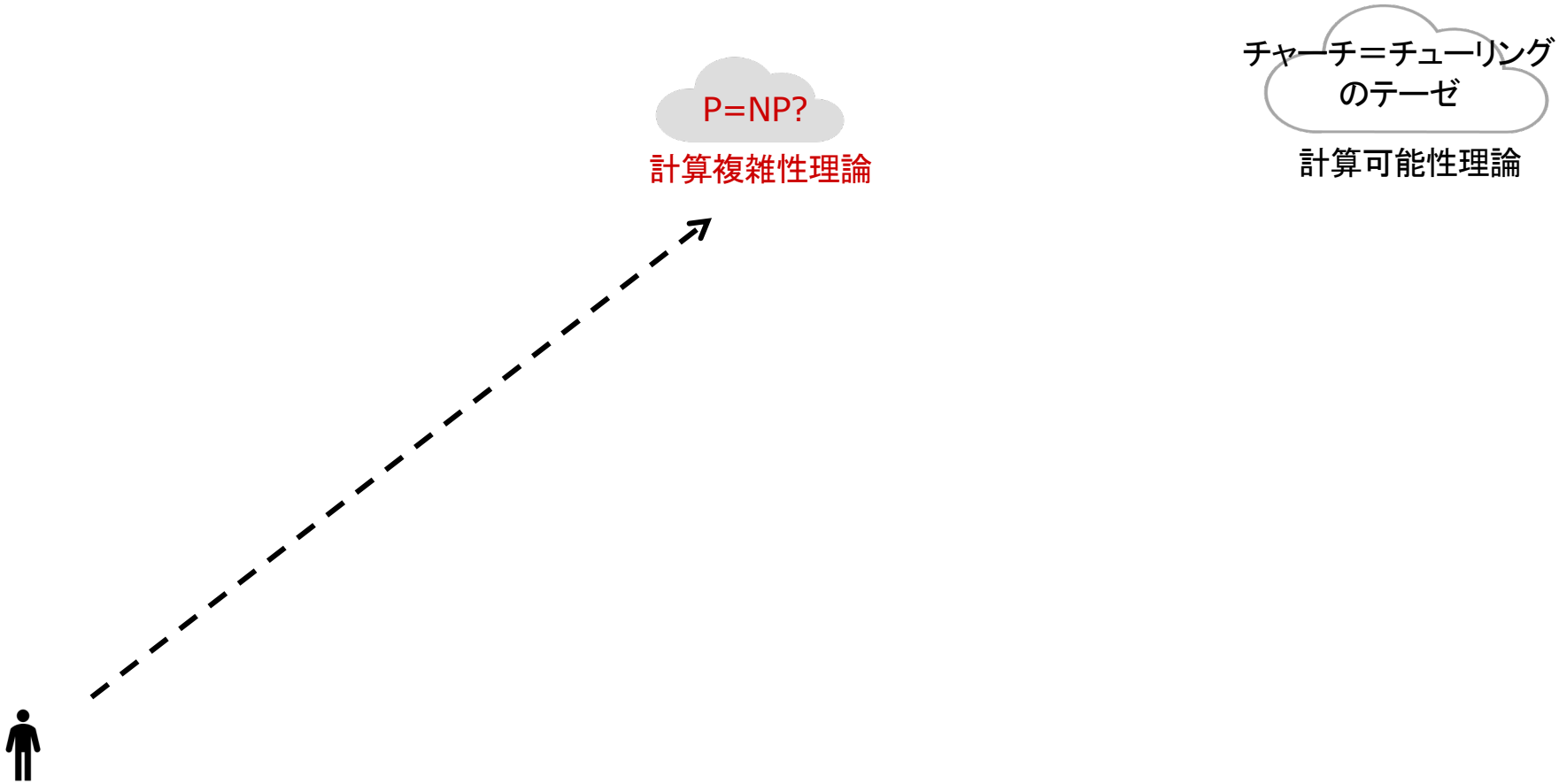
Scott Aaronson “P=NP?” https://goo.gl/XzekU1

ある数学的命題Sの真偽を決定する「純粋に機械的手続き」は存在しないにせよ、そのSが、ある制限長さn以下の証明を持つかどうかを決定する手続きは存在する。単純に、長さn以下の証明を全て枚挙して、それが命題Sの証明になっているかをチェックすればいい。

しかし、この方法は指数関数的な時間を必要とする。P=NP?問題は、こうした問題を解く「高速」なアルゴリズムがあるかを問うている。その意味では、 P=NP?問題は、ヒルベルトの問題提起の現代版だと考えることができる。

この問題が、明確に提起されたのは、1970年代の初めのことで、CookとLevinの仕事である。（現在のようなスタイルで、N = NP? 問題を最初に定式化したのは、バークレーのStephen Cookだという。1967年のことだ。） ただ、このことは、50年代に、ゲーデルもナッシュも気がついていた。

70年代



ただ、計算複雑性理論の基本問題である、「P ＝ NP ？ 問題」（それは、「難しい計算」が「易しい計算」には還元できないことを主張する）が未解決のままなこともあり、この理論と人工知能論との接点は、大きなものには広がらなかった。

計算複雑性理論の基本的クラス

NP問題

NPクラスの説明によく用いられるのは、素因数への分解問題である。ある自然数Nを「多項式時間」で素因数に分解するアルゴリズムは、現在では、まだ見つかっていない。（多分、そんな多項式時間アルゴリズムは見つからないだろうと感じている人が多いのだが、かといって、そのことが確定しているわけではない。）

一般の数Nではなく、条件を緩めて、Nは二つの素数の積であることがわかっていたとしても、Nが大きな数になると、Nを素因数に分解すること、すなわち、N＝pqであるような二つの素数 p, q を見つけることは、極めて難しくなる。

ただ、N, p, q が具体的に与えられているなら、「Nは、素数pと素数qに、素因数分解される」という主張は、簡単に検証できる。pとqを掛けて、その結果がNに等しいことを示せばいいのだから。掛け算は、もちろん簡単な計算で、多項式時間で終わる。

こうした、その問題を解くのが「多項式時間」で終わるかどうかはわからなくても、具体的な値で与えられたその問題の「答え」と言われるものが、本当に「正しい答え」であるかどうかを「多項式時間」でチェックできる問題のクラスを、NPという。素因数分解問題は、NP問題である。

（ちなみに、「答え」（なるもの）が与えられた時、それが「間違った答え」であるかどうかを「多項式時間」でチェックできる問題のクラスを、co-NP問題という。素因数分解問題は、co-NP問題でもある。）

こうしたことや、NP問題である素因数分解問題に現れる「難しさ」と「易しさ」の「非対称性」が、現在の公開キー暗号の基礎になっていることは、多くの人が既に知っていることかもしれない。

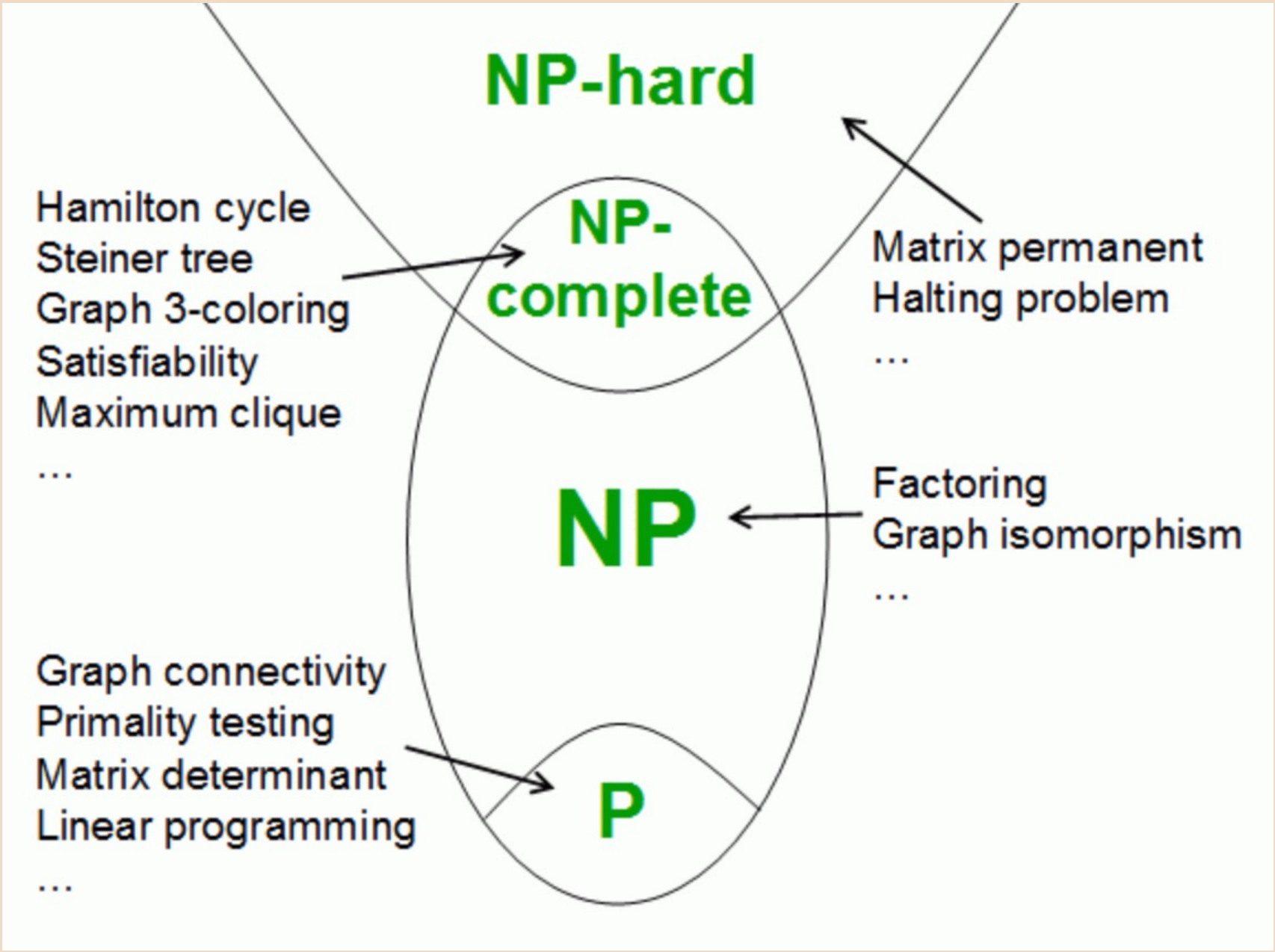
- **Denition. P** is the class of languages L $\subset \{0,1\}^*$ for which there exists a Turing machine M and a polynomial q so that for inputs x $\in \{0,1\}^n$, M terminate in at most q(n) steps and accepts if and only if x $\in$ L.
- **Denition. PSPACE** is dened like P, except we're limited by q(n) space, rather than time.
- **Denition. EXP** is dened like P, but we're limited by $2^{q(n)}$ time steps.

- **Denition. BPP** is the class of languages L $\subset \{0,1\}^*$ for which there exists a Turing machine M and a polynomial q so that for inputs x $\in \{0,1\}^n$, M terminate in at most q(n) steps and
  - if x $\in$ L then M accepts with probability > 2/3
  - if x $\notin$ L then M accepts with probability < 1/3

The constants 1/3 and 2/3 aren't important; we can amplify to make the success probability as high as we want by running the program a bunch of times and taking the majority result.

We can pull the randomness out of the denition of BPP by making M take an addition argument r, which can take values in $\{0,1\}^n$

- **Denition. BPP** is the class of languages L $\subset \{0,1\}^*$ for which there exists a Turing machine M and a polynomial q so that for inputs x $\in \{0,1\}^n$, M terminate in at most q(n) steps and
  - if x $\in$ L then M(x, r) accepts for at least 2/3 the possible values of r
  - if x $\notin$ L then M(x, r) accepts for at least 2/3 the possible values of r


NP-hard
NP-complete
NP
P
Hamilton cycle
Steiner tree
Graph 3-coloring
Satisfiability
Maximum clique
…
Matrix permanent
Halting problem
…
Factoring
Graph isomorphism
…
Graph connectivity
Primality testing
Matrix determinant
Linear programming
…

代表的な NP-Complete の例

- **Satisfiability**: the boolean satisfiability problem for formulas in conjunctive normal form (often referred to as SAT)
  - **0–1 integer programming** (A variation in which only the restrictions must be satisfied, with no optimization)
  - **Clique** (see also independent set problem)
    - **Set packing**
    - **Vertex cover**
      - **Set covering**
      - **Feedback node set**
      - **Feedback arc set**
      - **Directed Hamilton circuit** (Karp's name, now usually called **Directed Hamiltonian cycle**)
        - **Undirected Hamilton circuit** (Karp's name, now usually called **Undirected Hamiltonian cycle**)

https://goo.gl/56qLWk

代表的な NP-Complete の例

- **Satisfiability with at most 3 literals per clause** (equivalent to 3-SAT)
  - **Chromatic number** (also called the Graph Coloring Problem)
    - **Clique cover**
    - **Exact cover**
      - **Hitting set**
      - **Steiner tree**
      - **3-dimensional matching**
      - **Knapsack** (Karp's definition of Knapsack is closer to Subset sum)
        - **Job sequencing**
        - **Partition**
          - **Max cut**

https://goo.gl/56qLWk

PSPACE-complete problems

- Logic

  Quantified boolean formulas · First-order logic of equality · Provability in intuitionistic propositional logic · Satisfaction in modal logic S4 · First-order theory of the natural numbers under the successor operation· First-order theory of the natural numbers under the standard order· First-order theory of the integers under the standard order · First-order theory of well-ordered sets · First-order theory of binary strings under lexicographic ordering· First-order theory of a finite Boolean algebra· Stochastic satisfiability · Linear temporal logic satisfiability and model checking

- Lambda calculus

https://goo.gl/xZJYF5 https://goo.gl/hXveKm

PSPACE-complete problems

- Formal languages

  Word problem for context-sensitive language · Intersection emptiness for an unbounded number of regular languages · Regular expression star freeness · Equivalence problem for regular expressions · Emptiness problem for regular expressions with intersection. · Equivalence problem for star-free regular expressions with squaring.
- Quantified Boolean formulas
- Puzzles and games

https://goo.gl/xZJYF5 https://goo.gl/hXveKm

複雑さについての多様なアプローチ

コルモゴロフの複雑性

コルモゴロフの複雑性

計算機科学の分野では、「複雑さ」の尺度に、「コルモゴロフの複雑性」と言う概念がよく用いられる。それは、ある対象の「複雑さ」を、その対象を出力するプログラムの「大きさ」に置き換えて説明しようというアイデアに基づいている。

簡単なものを出力するプログラムは短く、複雑なものを出力するプログラムは長い。これは、プログラマーなら経験的・直感的に理解できることだと思う。

また、これもプログラマーならわかると思うのだが、同じ結果を返すその意味では同じプログラムでも、ある人のプログラムは冗長で長く読みにくく、ある人のプログラムは、コンパクトでわかりやすい。

同じ結果を返すいくらでも冗長なプログラムはかけるのだから、出力の「複雑さ」を、プログラムの「大きさ」に対応づけようとするのなら、その結果を返す「最も短いプログラム」の大きさを考えるのがいいことになる。ある対象の「コルモゴロフの複雑性」と言うのは、その対象を出力する「最も短いプログラム」の「大きさ」のことである。

もちろん、どの言語を選ぶか（Java, Python,Assembler, ...）、どのようなハードウェア・システム構成（GPUの有無、データベースの有無、...）を選ぶかで、この「最も短いプログラム」の「大きさ」は変化する。

ただ、プログラムの実行環境を固定すれば、その環境のもとでの「コルモゴロフの複雑性」は、その意味では相対的にではあるが、一意に定まるはずである。

繰り返しになるが、ある対象の「コルモゴロフの複雑性」と言うのは、その対象を出力するプログラムの大きさが最小の自然数である。ある対象を出力する、同じ機能を持つ多数のプログラムがあったとしても、そのプログラム・サイズからなる自然数の全体の集まりを考えれば、その中に、最小の自然数が、一つは存在するはずである。

実行環境を固定しても、実際には、ある対象の「コルモゴロフの複雑性」を具体的に「計算」することは難しいかもしれない。この辺りが、「コルモゴロフの複雑性」がわかりにくいと感じられるところかもしれないのだが。

ただ、ここは「コルモゴロフの複雑性」理解の上で、飛躍が必要なポイントでもある。重要なことは、具体的にその数字を示すことができなくとも、また具体的にそのプログラムを示すことができなくとも、最小のプログラム・サイズが存在するのは確かであると言うことである。

チャイティンの不完全性定理

チャイティンの不完全性定理

コルモゴロフは、旧ソ連の数学者で、「コルモゴロフの複雑性」が定式化されたのは、1960年代のこと。今から50年近く前の話だ。

「コルモゴロフの複雑性」が奇妙な性質を持つことを、驚くような形でまとめたのは、チャイティン(G. J. Chaitin)である。

「どのような特定のビット列をとっても、そのコルモゴロフの複雑性が L 以上であることを証明できないような数 L が存在する。」

これを「チャイティンの不完全性定理」という。

どう言うことか？

いくらでも複雑なものは存在する。どんな数に対しても、その数以上のコルモゴロフ複雑性を持つビット列は、無限に存在することを、我々は証明できる。ただ、特定のビット列が与えられた時、そのコルモゴロフ複雑性が L 以上であることを証明できない Lが存在すると言うのだ。

いくらでも複雑なものを、我々は、無限に考えることができる。複雑なビット列を考えて、それとは違うもっと複雑なビット列を考えて、もっと考えて、 …… これを無限に繰り返しても、具体的なビット列を考える限り、その複雑さは L を超えないと言うのだ。

まるで、孫悟空がいくら飛び回っても、お釈迦様の手のひらから飛び出すことができないように。複雑さを求めて、宇宙の果てまで来たつもりだったのに、そこには超えられない壁 L があることを見つけたと言うことだ。

「複雑さ」（我々が具体的に考えうる）には、乗り越えられない壁 L があるのだ。

そういうと、Lは、とんでもなく巨大な数だろうと考えたくなるのだが、Lは、驚くほど小さいらしい。

compositionality というのは、複雑なものが単純なものから構成されるという性質のことだと書いたが、単純なものから複雑なものを構成するルールをが、Nビットで記述できるとすると、複雑さの壁 L は、

$$L <= N + 2359$$

程度だという。正確に、Lの値を求めることはできないが、あるコンピューター科学者は、数キロバイトだと推測している。

小さい！

特定のビット列を出力するプログラムは、そのビット列がどんなに複雑でも、数キロバイトに収まるということで、皆さんが普段作っているプログラムが、数キロバイトになるはずだということではないので、ご安心を。

チャイティンは、こういう言い方もしている。

「あるプログラムを、それより小さいサイズのプログラムが同じ出力を行うことがない時、"エレガント"と呼ぶことにしよう。あるプログラムが "エレガント" であることを、我々は証明できない。」

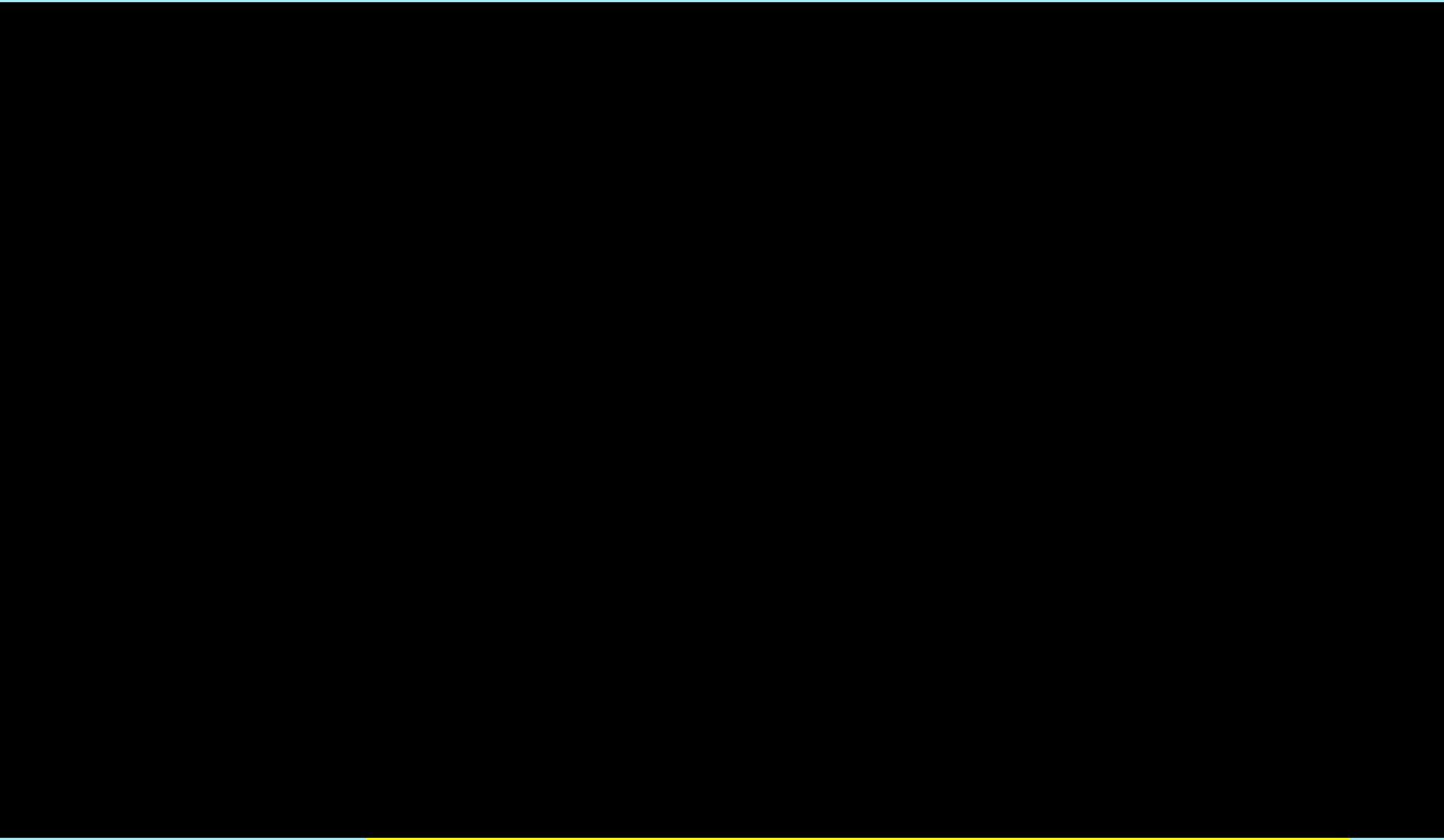
実際に、小さなプログラムでも、複雑なビット列を生み出すことができることの例を二つ紹介しよう。

一つ目は、マンデルブローのフラクタル図形の例である。この画像は、1ピクセルあたり24bitカラーを用いていて、全体で、162万bitあるのだが、これを生成するプログラムは、本当に小さい。



「複雑系」の議論では、複雑さの例として取り上げられることがあるフラクタル図形だが、コルモゴロフの複雑性の尺度から言うと、この図形の複雑さは、驚くほど低い。

二つ目の例は、DemoSceneのコンテストで優勝した動画。見知らぬ惑星の描写だが、音楽含めて、プログラム・サイズは、たったの4Kだという。信じられない！



https://www.youtube.com/watch?v=I5CTFMuFvb0

我々に手の届かない有限について
Busy Beaver 問題

Busy Beaver 問題

Turingマシンの「停止性問題」は、あるTuringマシンが答えを出して停止するのか否かを問題にするのだが、この「忙しいビーバー」問題は、Turingマシンが停止するまでのステップ数の最大値を求めるという問題。

Turingマシンは、内部状態を持つ。状態Sでヘッドのあるマス目の値が、'0'か'1'かに応じて、状態をSからS'に変更し、'0'か'1'を書き込んで、ヘッドを右（R）あるいは左(L)に移動する。こうした命令の集まりが、Turingマシンの「プログラム」になる。

Turingマシンのプログラムの例を示す。この表の、例えば、最初の State Aの行で、On '0' の欄の 'B1R' は、「状態Aで、ヘッドが'0'の上にあるなら、状態をBに変えて、1を書き込んで、ヘッドを右（R）に移動する」という命令を表す。

State	on 0	on 1	on 0			on 1		
			Print	Move	Goto	Print	Move	Goto
A	B1R	D0L	1	right	B	0	left	D
B	C1R	F0R	1	right	C	0	right	F
C	C1L	A1L	1	left	C	1	left	A
D	E0L	H1L	0	left	E	1	left	H
E	A1L	B0R	1	left	A	0	right	B

On '1' の欄の 'D0L' は、「状態Aで、ヘッドが'1'の上にあるなら、状態をDに変えて、0を書き込んで、ヘッドを左（L）に移動する」という命令を表している。

この表のプログラムで定義されるTuringマシンは、A, B, C, D, E, F の6つの状態を持っている。このサンプルの、プログラム自体は、単純である。もちろん、マシンは停止状態HALTを持たないといけないのだが、ここでは、HALT状態Hは、数にカウントしていない。Dの行に出てくる 'H1L' が停止の命令になっている。）

「忙しいビーバー」問題は、状態の数がN個のTuringマシンを全部考えて、全て0で埋まっているテープから初めて、一番長いステップで停止するTuringマシンとそのステップ数を求めよという問題である。その最大値をBB(N)で表すことにすると、次の結果が知られている。

BB(1) = 1.
BB(2) = 6.
BB(3) = 21
BB(4) = 107

実は、まだ、誰も、BB(5)の値を知らない。

BB(5) ≥ 47,176,870

というのはわかっている。（誰かが、状態の数が5つで、47,176,870ステップで停止するTuringマシンを作ったということ。ただ、それが「最大」だということはわかっていない。）

そんなものかと思われるかもしれないが、雲行きが怪しくなるのは、BB(6)あたりから。

BB(6) ≥ 7.412 x (10の36,534乗）

だという。36,534桁の数字になる。先のBaezのblogに、「具体的に」この数字が紹介されている。延々と数字が続く。3万6千桁以上の数字が。

よくそんなことわかるなと思われるかもしれないが、それは、先にも述べたように、実際にそういう6状態のTuringマシンを構成して見せたからだ。実は、先に紹介した、6個の状態と12個の命令からなるTuringマシンが、そういう動き方をするのだ。

「プログラム自体は、単純である。」と先に書いたのだが、正しくないかもしれない。このプログラムは、えんえんと走り続け、そして、最後に停止する。（停止すること自体、不思議な気がする）

741207853509495610174172561144604969718281611695298008925669010951656624280328485493565509745496832570980660176344529980240910175257246046044979228025757711518548052087650589935156483217413541197777965279255493532447649712935848974978461539867784215790591584199376184970716056712502662159444630412079992352830139286757106976976266378010157256661988247506945421793112446656331449750558811894710601772365595995396507670767065001451170294752766860167506529522954129044871107849518263169509747269711158732776867610634089559834790714490552734463125404673708090108604513212129769190196259350728934650321231429040253205457633515626107868771760119916923828746803714584592161274169396551793596257979821625066031449422781829328977925461473293508048670145466821939225145869908038228128045266110256571782631958926898525694689966695874227519616911181790608398001974214915398771591696883364753477480074875777666112880815431005997890623859440699663891591940342058445345135951600168916065895276541430702668840252531350653890897051982632630385938083660639947985722351182179370081120817877269116559398341767052162655917201202433328300323752878230642831971805072395297353229551731048371021811597619301167315826868031 1

735322955173104837102181159761930116731582686803119630571008041911930477971579672785029529559439797294500432643483372677378872480519292318129075141594300170423748519482724090149197762230095180896645720799274350771114821420869806320389838466348644400634378985820511533007233636175063664244348363429381716865277675927173868435130844307605385977884978545703928873633762169439444600268593765042490439747070469396499307353236961551408770635423051623551580205020464330288028606243330668268170811900863960680544921270550820866591183165171362199062268022146340355100698032539208500321737935916572824167109243921796327708886988064622392869223657202340493532622931976010973933691955155585614971701393657083510829337138019755841416703048425473095781543947284695305578910483032968871052034458997996570053793215656951656746253672334104502853352759013244455004163729329181785691615066366432555395455966924963740865923479058518089001727259875665774273483385120744001467195339376792213348011167418166728360082829942361956450241322000295203110123701834947807654204715508724845292827346100146348547365089126533446859392192538154695164187041834978200787084142435296024 6

816219279435128009691488333367252441723619461885472096381488087746264915498258124361228833219320352241878334479833109122807808425070272194370533803098155762074363594124056079914285825861353250006004503404457075561484235380160575513851472863744453819991270653752636827482388619627545586769702982563550215791905943470016781247941475207377805457107252380926307057894825122170537840420055751812318342976374391628221317569903581457033268409573939140317537929519452225728328543766673545899812218722084639419217330239037159748031355083246976463886069438573523920879420538715366934472880272772745254215764827226580772102826496399117753878844601174815425740209860471059749736357781622490624052946817662800124337027642430572009172724680494845807607875336391296355953749364637564991523417213639553068550050631478405859742434139260653244354541439306503595217538645638222669677018885647203581909266795843083183075450785277713783211861708706612681436736614104408791294005647913540430281084317983076118608172715678564098233869131431441387859096996327035057252704630665023999285518292849124642753224570810976792933497725693910894339658778178382786680971310533947980194252766851530607523746692117596241149131822801952

28443629054406029354014078168448839468543109777746497134194328220740395976456632163669113880556744440040338242074160211898209536897949768268405300638550209609958621490671271332429239552166892883811184405888820990463604425076520669497073794980146362709477533118591401481473656166664409698471099509772674271268524194763314787756784416422586929186303999309479972891692726739231712531500339659100715122651178203821487177649041575923087270542299624201754570706993341240356064699636293209512874013786250682473887757931001901807158800515178567563191206326463879171290239696789072427830303321073398269847363456290199268793655334873976194500237322039977440978878227032599708324913637134795947392057672257001889829885987903466227757446048410092755426060057477348984785786907788507119614119876333605601699259296191798210522981667181477607820683943238312997333502226273311447560030346344769138042632040645897100672747110856632877598317707068109285992387448288963033783848284343008603095759504211313687275146815571999153003835709371811628295885386861489714678254967080333475258187514533923483088945298272830364477058058993426870144942681269232766983593731414821

4467492875119988039620924341039458961770757345685
6454301520275813300267434717502921849792945757378 6
6546765128985326270025306439142280125170385670430 4
3993367412997435263933332827902185399873234049339 1
5243986633960766937577765446089358460929132081948 2
3545029493121851964625769147302478463529246280565 4
6056581254571018956482544451653310465454962630777 4
1375950179168158540681999239199541083222930573503 1
7974307367947840165992935953268841291762992863264 6
2353158681102294807472460144080798083883059422402 0
7030904215718776078177940150483268879448134664598 9
9784894146719136782011032591772330616588698650629 4
3183141236334841851779088120360233282957052325831 7
1794949717135562421748095886304059101561741816275 0
6372430552440509130586107235520485592847579364235 7
6024628034664212377839601949771029506076828600446 0
9230861730073523130319643352345732610347023685817 8
2841443182830031237366080354236839216889742383761
8082177034765392656093836807671696102263353151287 2
8850418398744182010857362109000196186093896160246 0
2088568776363979501033526558876797002408567338277 0
4944534279588887636004528374067296959992161212008 8
1908843324216508529595491070765557857624369203455 1

省略

46076619939139036787876060299301996426764725427403
64011297421238171255442319681637731281152043413431
85060894391599524452878044334616995055050095307491
88403737341503747816446538950384710455426239295288
28998531209709108913722340683342671325423513366061
63474488522182030819462026736771976453958939935828
32050459497790182712782471389976977879426526570877
30339646201949991550918186788970663139748645058559
72718469140463791494759075349756092667385993283854
20778335173462088786041537266587943086744729710814
62579154103447702796046390902553975412328182470276
13052411058118309311147944260478608

steps and then halts!

有限だが計算できない数の発見について

最近になって、「忙しいビーバー」 Busy Beaver 問題で、とても重要な発見があった。

何がわかったかを述べる前に、少し、状況を整理しよう。

Busy Beaver 問題というのは、N個の状態を保つTuringマシンが、実際に計算可能な最大ステップ数 BB(N) を求めよという問題だ。

まず、N個の状態を持つTuringマシンが何個あるか考えよう。これは、昨日の遷移表を眺めていれば、簡単にわかる。

'B1R' や'H1L'といった「命令」の数は、一文字目が N+1個（状態の数＋停止状態）、二文字目が2個（ 1 か 0 だから）、三文字目が2個（RかLだから）の可能性があるから、全部で、(N+1) x 2 x 2 通りの組み合わせがある。

Turingマシンは、これらのN x 2 個の命令 (一つの状態について、1の場合と 0の場合について、実行する命令を置いたものだから)から構成される。

だから、N個の状態を持つTuringマシンの総数は、2N個のマス目を、4(N+1)個のもので埋めればいいので、4(N+1) の2N乗だということがわかる。

これは、大きな数になるかもしれないが、「有限」な数だ。だから、これをかたっぱしから試していけば、「原理的」には、BB(N)は求まるはず。

と思うかもしれないが、この手続きはうまくいかない。だって、試すべきマシンの数が「有限」でも、「無限」ループにおちいるマシンにぶつかれば、この手続きは「有限」の時間では終わらないから。

ただ、こうしたループの「落とし穴」ではない、もう一つの問題がある。それは、状態の数Nやチェックすべきチューリングマシンの数と比べて、BB(N)が、きわめて急速に巨大な数に膨れ上がるということだ。（単純なループにはまるのと、答えを求めて、えんえんと計算を続けるのは、多分、別のことだ。BB(N) は、そうした最長の数を求める。）

先に、BB(6)が、36,534桁の数字より大きいという話をした。2014年に、Wythagoras（怪しい名前だ）は、BB(7)について、次のような評価を示した。右側の数は、とても巨大な数だが、「有限」な数字である。

$$\mathrm{BB}(7) > 10^{10^{10^{10^{10^7}}}}$$

冒頭に述べた「Busy Beaver 問題での重要な発見」の話をしよう。

今年（2016年）の4月に、Adam Yedidia と Scott Aaronson は、BB(7910)が、普通の集合論の枠組みの中では決定できないことを証明した。

"A Relatively Small Turing Machine Whose Behavior Is Independent of Set Theory"
http://arxiv.org/pdf/1605.04343v1.pdf (Scottの次のblogが参考になる。
http://www.scottaaronson.com/blog/?p=2725)

それ以来、BB(N)が計算不能になる、より小さいNの探索が、インターネット上で活発に行われ、この "logic hack" で、つい二週間ほど前に、BB(1919)も計算不能であることが示されたのである！

確かに、僕が生きているうちは、BB(10)の値も決まらないだろうと思って、ほぼ、間違いないので、BB(7910) とか BB(1919)とかは、とんでもなく大きな数である。ただ、これらの数字は「有限」で、しかも、BB(N)自体は、とても明確に定義されたものである。

要するに、「有限」の値を持つのだが、どういう数学的な手段を持ちいても、我々が知り得ない数字があるということ。さらに、その限界が、7910とか1919といった、比較的小さな数字で特徴付けられるというのが、驚きである。

チャイティンとゲーデル

チャイティンとゲーデル

1974年のことだ。27歳のチャイティンは、ドキドキしていた。自分のアイデアをゲーデルに話してみようと思ったからだ。

訪問先のワトソン研の電話を借用して、彼は、ゲーデルに電話をかけた。なんと、ゲーデルが電話に出た！

「せ、先生、僕は、先生の不完全性定理に魅了されています。ぼ、僕は、ベリーの逆理に基づいた、不完全性定理の新しい証明を行いました。そのことでお話がしたいのですが」
「君が、何の逆理を使おうが、僕にはどうでもいいのだが。」

ゲーデルは、不完全性定理の証明に「嘘つきのパラドックス」を使っていた。

「は、はい。おっしゃる通りです。でも、僕の証明は、不完全性定理の情報理論的な観点からの解釈を与えるものだと思います。そのことで、ぜひとも、先生にお話ししたいのです。ぜひ、先生のお考えを伺いたいのです。」

「君の論文を僕に送りなさい。読んでみるから。数週間後に、もう一度電話して。予定が取れたら会えるかもしれない。」

短い電話だったが、チャイティンは、跳び上がるほどに喜んだ。証明のアイデアを得たのは、1970年だったが、まだ短い要約しか発表していなかった。しかし、幸運なことに、この1974年には、チャイティンはこのテーマでの最初の重要な論文を完成させたばかりだった。チャイティンは、すぐに論文を封筒に入れ、ゲーデルに郵送した。

数週間後、チャイティンは、再びゲーデルに電話した。何と、ゲーデルは、会う約束をしてくれたのだ！　チャイティンは、どんなに嬉しかったことだろう！　ゲーデルがいるプリンストンまで、汽車でどうやっていけばいいのか、浮き浮きとした気分で計画を考えた。

その日が来た。

その日は、雪だった。あたり一面、数センチの雪で真っ白だった。ただ、チャイティンがゲーデルを訪問することを妨げるものは、何もないのは、明らかだった！

列車に乗るために出かけようとしたその時、電話がなった。ゲーデルの秘書からだった。

「先生は、健康に特別に気を使っています。この雪で、先生は、研究所には行かないそうです。」

チャイティンのアポは、ドタキャンされたのだ。
可哀想なチャイティン！　その後、チャイティンがゲーデルに会うことは、二度となかった。

"The Berry Paradox" https://arxiv.org/pdf/chao-dyn/9407003v1.pdf から。一部、丸山が脚色。

教訓1：健康に気を使うのはいいけど、体に毒なものを恐れて、最後は栄養失調でゲーデルは死んだらしい。健康志向はほどほどに。

教訓2：情報科学にとって、ゲーデルの定式化より、チャイティンの不完全性定理の方が、ずっと役に立つ。

教訓3：若者をなめるんじゃない。

Part IV

量子複雑性理論

Agenda Part IV
量子複雑性理論

- 量子複雑性理論
  - Feynman
  - Deutsch
  - BQP Bernstein と Vazirani
  - Quantum Complexity Theory
  - Shor
  - Extended Church-Turing Thesis
  - NP-complete Problems and Physical Reality
- 物理学と量子情報理論

「計算可能性理論」が、大きな転機を迎えるのは、1980年代に入ってからだ。

ファインマンが、コンピュータでは量子力学の法則に従う自然のシミュレートができないことに気づく。彼は、自然のシミュレートが可能なコンピュータは、量子力学の法則に従ったコンピュータでなければならないと主張する。

この指摘が、「量子コンピュータ」研究の始まりである。

数学的体系と同様に、自然もまた、我々の認識の対象である。数学だけでなく、物理学もまた、我々の認識の可能性と限界について、強い関心を持っているのだ。

ドイッチェは、ファインマンの考えを受けて、チューリングマシンの量子版を構成し、こうした量子チューリング・マシンで計算可能なものが計算可能であるとする。これを、計算可能性についての「チャーチ＝チューリング＝ドイッチェのテーゼ」という。

一見すると、同じような定式化に思えるのだが、本質的な違いがあるのだ。

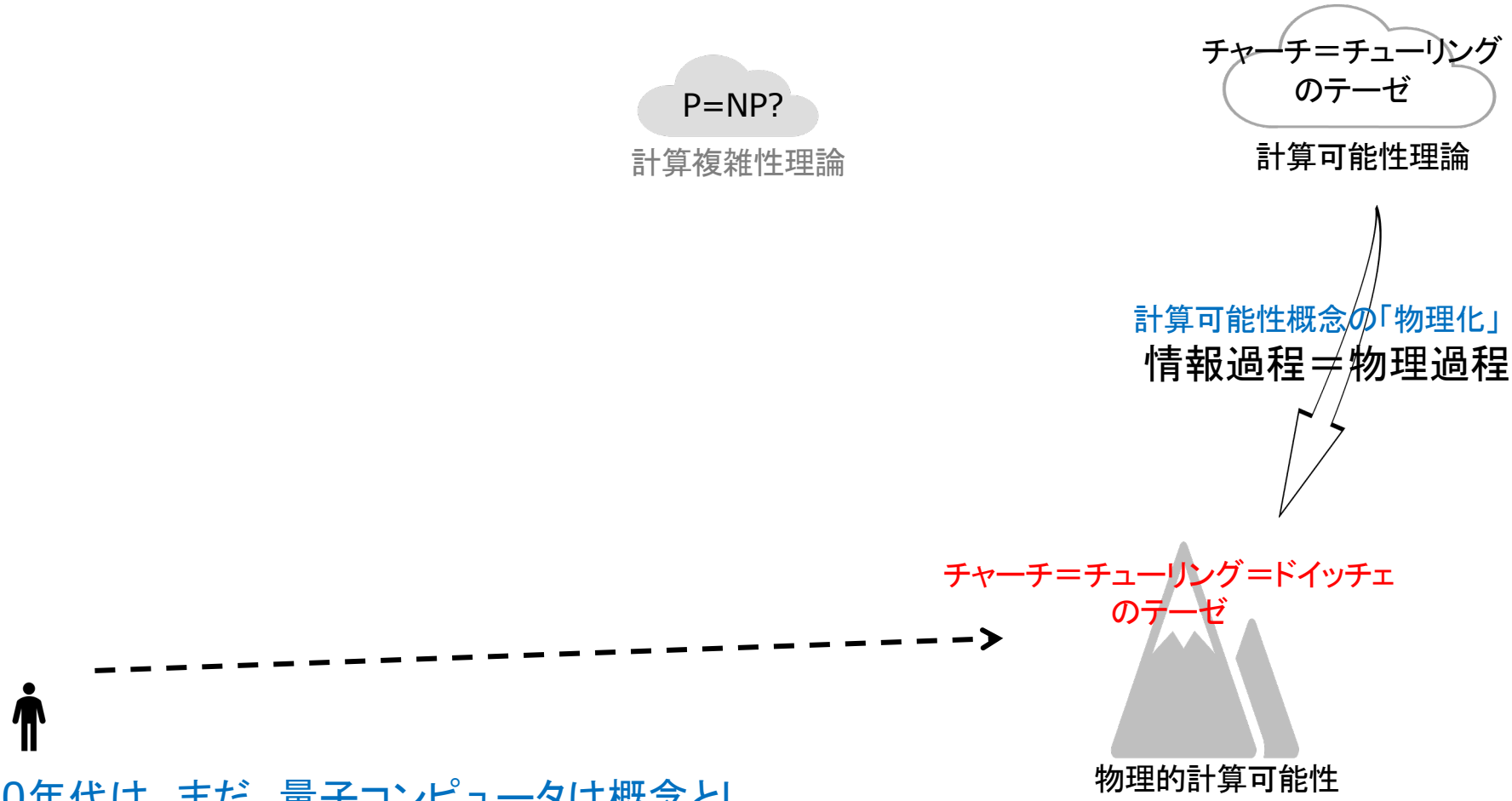
「チャーチ＝チューリングのテーゼ」は、抽象的・形式的・数学的な「計算可能性」の定義についての提言なのだが、「チャーチ＝チューリング＝ドイッチェのテーゼ」は、「計算可能性」が、実在的・物理的に定義されねばならないと主張しているのだ。

「チャーチ＝チューリングのテーゼ」は、いわば、遠くの雲の上に抽象的に存在する原理だったのだが、「チャーチ＝チューリング＝ドイッチェのテーゼ」は、地上に降りた現実の物理的原理だ。

重要なことは、こうした「計算可能性」概念の「物理化」の背景にある思想である。それは、情報過程が、けっして抽象的なものではなく、物理的なものに支えられた物理過程に他ならないという考えである。

ただ、80年代は、まだ、量子コンピュータは概念としてしか存在していなかった。「計算可能性」の物理化という画期的な変化も、まだまだ、抽象的な議論だった。

80年代



80年代は、まだ、量子コンピュータは概念としてしか存在していなかった。「計算可能性」の物理化という画期的な変化も、まだまだ、抽象的な議論だった。

Feynman

"Simulating Physics with Computers”
1982年

https://goo.gl/ueVbdp

「計算可能性」の理論として出発した「複雑性理論」が、質的な深化を果たすのは、1980年代になってからである。

大きな転回点になったのは、1982年のファインマンの論文 "Simulating Physics with Computers" https://goo.gl/ueVbdp である。

この論文は、複雑性理論の進化の重要な画期を与えるとともに、現在の量子コンピュータのアイデアの出発点ともなる重要な論文である。（同時に、現在の量子コンピュータ技術の到達点は、ある意味、この論文への「回帰」として特徴付けられるのは、興味ふかいことである。

ファインマンは、次のように述べる。

「コンピューターが、正確に自然と同じように振る舞う、正確なシミュレーションが存在する可能性について話そうと思う。」

「それが証明されて、そのコンピュータのタイプが先に説明したようなものであるなら、必然的に、有限の大きさの時空の中で起きる全てのものは、有限な数の論理的な操作で正確に分析可能でなければならないことになるだろう。」

「量子論的なシステムは、古典的な万能計算機で、確率論的にシミュレートされるだろうか？」

「 別の言い方をすれば、コンピューターは、量子論的なシステムが行うのと、同じ確率を与えるだろうか？　コンピューターを今まで述べてきたような古典的なものだとすれば（前節で述べたような量子論的なものではないとすれば）、また法則はすべて変更されないままで、ごまかしもないとすれば、答えは明らかにノーである。」

「それは、新しいタイプのコンピューター、量子コンピューターで可能になるだろう。」

「 私が理解する限りでは、それは量子論的なシステムによって、量子コンピューターの要素によって、シミュレート出来るようになることは、いまや、明らかになった。それはチューリング・マシンではない。別のタイプのマシンである。」

Deutsch

"Quantum theory, the Church-Turing principle and the universal quantum computer"

1985年

https://goo.gl/PVHGKa

ファインマンの「量子コンピュータによる自然のシミュレーション」という刺激的な問題提起を受け、それを「計算可能性理論」の中心的な原理である「チャーチ＝チューリングのテーゼ」との関連で整理しなおして「計算可能性」を再定式化したのは、ドイッチェである。

1985年の論文 "Quantum theory, the Church-Turing principle and the universal quantum computer" https://goo.gl/PVHGKa で、ドイッチェは、「チューリング・マシンのクラスの量子論的一般化である計算機械のクラス」＝「万能量子コンピュータ」を構成して見せる。

彼は、「チャーチ＝チューリングのテーゼ」を次のように捉える。

『「計算可能と自然に見なされる関数」は、全て、万能テューリング・マシンで計算可能である。.』

ただ、彼は、この「チャーチ＝チューリングのテーゼ」の基礎には、暗黙の物理学的主張がある」という。これが、この論文の眼目である。

彼は、彼が構成して見せた「万能量子チューリング・マシン」＝「万能計算機械」を用いて、「チャーチ＝チューリングのテーゼ」を書き換える。

チャーチ＝チューリング＝ドイッチェのテーゼ

「この物理学的主張は、次のような物理学的原理として、明確に表現することが出来る。」

『有限な方法で実現可能な物理システムは、有限な手段によって操作される万能計算機械のモデルで完全にシミュレート可能である。』

これを、「チャーチ＝チューリング＝ドイッチェのテーゼ」という。

ドイッチェは、論文で、これらの原理が、次の熱力学の第三法則に似ていること注意を向けているのは、興味ふかい。

『どのような有限のプロセスも、有限な手段実現可能な物理システムのエントロピーあるいは温度をゼロにはできない。』

形式的・数学的で抽象的な「計算可能性」の原理だった「チャーチ＝チューリングのテーゼ」は、ここに「チャーチ＝チューリング＝ドイッチェのテーゼ」として、実在的な過程としての「計算」の原理として「物理化」されることになる。

「計算可能性」の原理の「物理化」は、関連する諸科学に、一つのインパクトをもたらすことになる。なぜなら、それは、それまで別々の学問の対象だった「計算過程」「物理過程」「情報過程」（それぞれ、数学・物理学・コンピュータ科学が扱っていた）に対して、「計算過程」＝「物理過程」＝「情報過程」という、強力な三位一体の図式が、しかも、物理学の優位の元に生まれたことを意味することになるからである。

こうした動きのすぐれた解説は、彼の死後に発刊されたファインマンの最後の著作である "Feynman Lectures on Computation" https://goo.gl/PyqgyT の第5章 "Reversible Computation and the Thermodynamics of Computing" と第6章 "Quantum Mechanical Computers" をみるのがいい。30年前のものだが、少しも古くない。

ただ、「チャーチ＝チューリング＝ドイッチェのテーゼ」は、その認識論的意味を考えると、本当は、なかなか手強いものだ。その辺の議論は、あとで振り返ることがあると思うが、ニールセンの次の投稿を見て欲しい。“Interesting problems: The Church-Turing-Deutsch Principle” https://michaelnielsen.org/blog/interesting-problems-the-church-turing-deutsch-principle/

BQP
Bernstein と Vazirani

“Quantum Complexity Theory”
1993年

https://goo.gl/3y4RSf

Just as the theory of computability had its foundations in the Church-Turing thesis, computational complexity theory rests upon a modern strengthening of this thesis, which asserts that, any "reasonable" model of computation can be efficiently simulated on a probabilistic Turing Machine (an efficient, simulation is one whose running time is bounded by some polynomial in the running time of the simulated machine). For example, computers that can operate on arbitrary length words in unit, time, or that can exactly compute with infinite precision real numbers are unreasonable models - since it seems clear that they cannot be physically implemented. It had been argued that the Turing Machine model (or the polynomial time equivalent cellular automaton model) is the inevitable choice once we assume that we can implement only finite precision computational primitives. Given the widespread belief that NP ≠ BPP, this would seem to put a wide range of important, computational problems (the NP-hard problems) well beyond the capability of computers.

However, the Turing Machine is an inadequate model for all physically realizable computing devices for a. fundamental reason: the Turing Machine is based on a classical physics model of the Universe. whereas current physical theory asserts that, the Universe is quantum physical. Can we get inherently new kinds of (discrete) computing devices based on quantum physics?

The first indication that such a. device might potentially be more powerful than a probabilistic Turing Machine appeared in a paper by Feymnan [Fe82] about a decade ago. In that paper, Feynman pointed out a very curious problem: it, appears to be impossible to simulate a general quantum physical system on a probabilistic TM without an exponential slowdown. The difficulty with the simulation has nothing to do with the problem of simulating a continuous system with a. discrete one - we may assume that, the quantum physical system to be simulated is discrete, some kind of a quantum cellular automaton. In view of Feynman's observation, we must reexamine the foundations of computational complexity theory, and the complexity-theoretic form of the Church-Turing thesis, and study the computational power of computing devices based on quantum physics.

In this paper, we prove the existence of a universal quatum Turing Machine whose simulation overhead is polynomially bounded. In full generality, on any given input a quantum TM produces a random sample from a probability distribution. We say that quantum TM T' simulates T with accuracy c, if on every input z, T' outputs a sample from a distribution which is within total variation distance c of the corresponding distribution for T. We prove that there is a universal quantum TM, which takes as input the description of a quantum TM T. time t, and input, z, and simulates T(x) for time t with accuracy ϵ. The slowdown is polynomial in t and $1/\epsilon$.

In this paper, we present, the first evidence that, quantum TMs might, be more powerful than classical probabilistic TMs. We prove that there is an oracle relative to which there is a language that can be accepted in polynomial time by a quantum TM but cannot be accepted in $n^{O(\log n)}$ time by a bounded-error classical probabilistic TM. A more careful construction exhibits an oracle relative to which quantum polynomial time is not even contained in the class Arthur-Merlin where the verifier has $n^{O(\log n)}$ time (see [BMW] for a definition of this class).

BQP

Let BQP (bounded-error quantum polynomial time) be the class of languages that are accepted by a polynomial time quantum TM with error probability at most 1/3. It is not hard to show that, BQP $\subseteq$ PSPAC'E. Therefore, we cannot, hope to prove that BPP $\subset$ BQP without resolving the longstanding open question BPP $\neq$? PSPACE. In fact, Valiant [Va92] recently pointedout to us that, the above result can be strengthened to say that BQP $\subseteq$ #P.

Controlling Error

- In the early days of quantum computing (the ’80s, say), skeptics said quantum computation was ‘just’ another form of analog computation (a machine manipulating real-valued quantities), and subject to the same limitation: perturbative noise that could erase information and throw off a computation by successively accumulating errors.
- This criticism ignored or glossed over an important insight (explained in Bernstein-Vazirani ’93): unlike other types of transformations used in classical analog computers, quantum operations cannot amplify error except by directly introducing more of it!

- Bernstein and Vazirani observe that the total error is (in an appropriate sense) at most the ‘sum’ of the errors in all preparation and gate components. Thus, if engineers can produce circuit elements with error 1/t we expect to be able to faithfully execute computations on quantum circuits with size on the order of t.

Threshold Theorem

- This idea was improved upon substantially. In a sequence of papers, work by Aharanov, Ben-Or, Knill, Laflange, Zurek and others culminated in the 1996 'Threshold Theorem', which showed how to use 'hierarchical coding' ideas to build quantum circuits for arbitrary BQP computations, which would remain reliable even if each gate was subjected to a sufficiently small (but constant) rate of error. This result was analogous to (but more complicated than) earlier work by Von Neumann on fault-tolerant classical computation.

Quantum Complexity Theory

"Quantum Complexity Theory"
Bernstein & Vazirani
1997年

https://goo.gl/jdSS91

Abstract

In this paper we study quantum computation from a complexity theoretic view point. Our first result is the existence of an efficient universal quantum Turing Machine in Deutsch's model of a quantum Turing Machine. This construction is substantially more complicated than the corresponding construction for classical Turing Machines - in fact even simple primitives such as looping, branching and comp osition are not straightforward in the context of quantum Turing Machines. We establish how these familiar primitives can be implemented and also intro duce some new purely quantum mechanical primitives such as changing the computational basis and carrying out an arbitrary unitary transformation of polynomially bounded dimension.

We also consider the precision to which the transition amplitudes of a quantum Turing Machine need to be specified. We prove that O(log T) bits of precision suffce to support a T step computation. This justifes the claim that that the quantum Turing Machine model should be regarded as a discrete model of computation and not an analog one.

We give the first formal evidence that quantum Turing Machines violate the modern complexity theoretic formulation of the Church-Turing thesis.

We show the existence of a problem relative to an oracle, that can be solved in polynomial time on a quantum Turing Machine, but requires super polynomial time on a bounded error probabilistic Turing Machine and thus not in the class **BPP** .

- The class **BQP**, of languages that are efficiently decidable with small error probability on a quantum Turing Machine, satisfies: $\mathbf{BPP} \subseteq \mathbf{BQP} \subseteq \mathbf{P}^{\#\mathbf{P}}$.

Definition of BQP

- **Definition. BQP** is the class of languages L $\subset \{0,1\}^*$ for which there exists a uniform family of polynomial-size quantum circuits $\{C_n\}$ over some basis of universal gates and a polynomial q so that for all n and inputs x $\in \{0,1\}^n$
  - if x $\in$ L then $C_n(|x> |0>^{\otimes q(n)})$ accepts with probability > 2/3
  - if x $\notin$ L then $C_n(|x> |0>^{\otimes q(n)})$ accepts with probability < 1/3

Since circuits have to pre-specify the input size, so we need a circuit for each input size n. By uniform, we mean that there is a classically ecient algorithm to produce C_n given n.

- We define the class **EQP**, exact or error free quantum polynomial time, as the set of languages which are exactly accepted by some polynomial time QTM. More generally we define the class **EQTime** (T(n)) as the set of languages which are exactly accepted by some QTM whose running time on any input of length n is bounded by T(n).
- More generally, we defune the class **BQTime** (T(n)) as the set of languages which are accepted with probability 2/3 by some QTM whose running time on any input of length n is bounded by T(n).

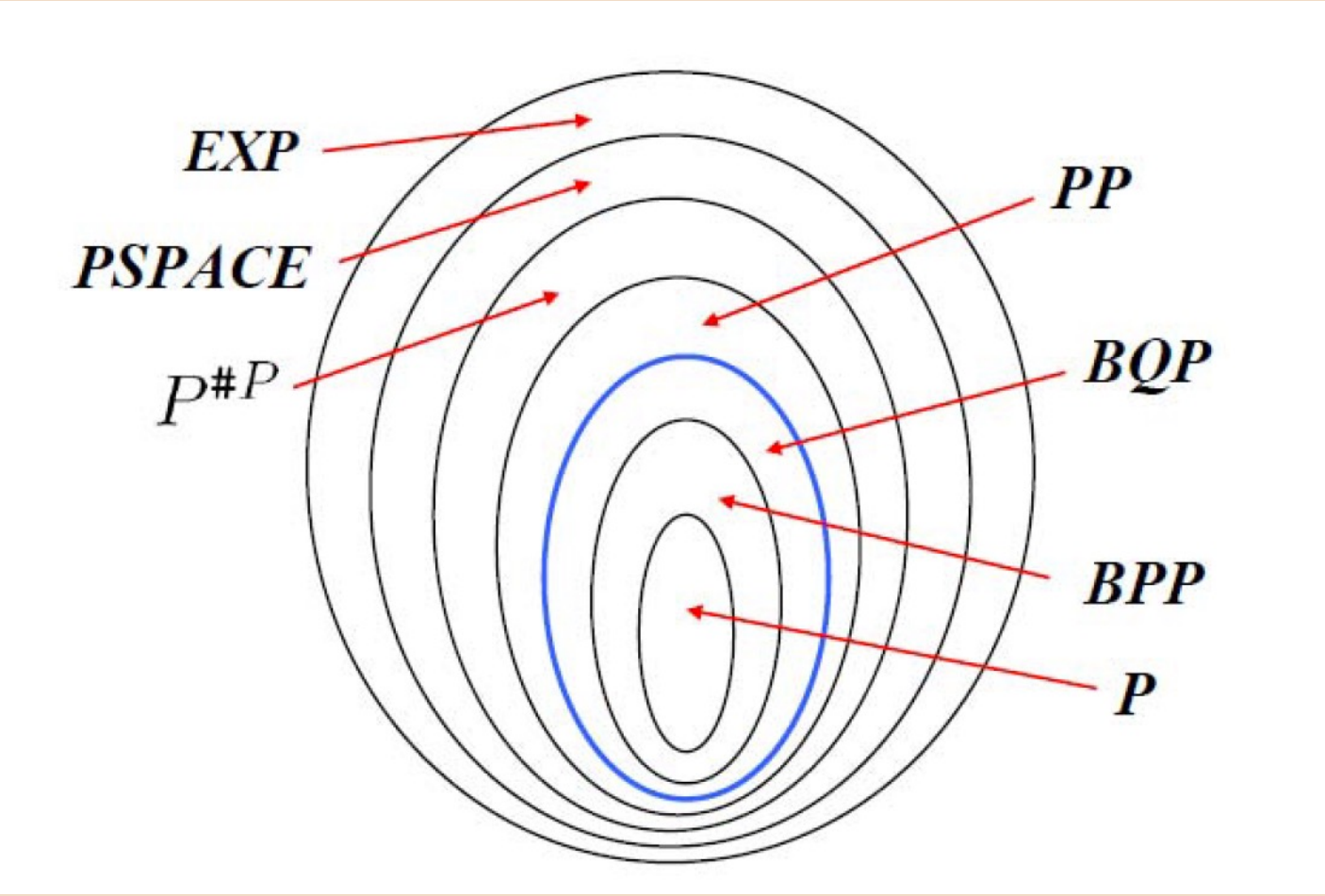
Upper and lower bounds on the power of QTMs

- Theorem 1 **P ⊆ EQP**
- Theorem 2 **BPP ⊆ BQP**
- Theorem 3 **BQP ⊆ PSPACE**
- Theorem 5 **BQP ⊆ $P^{\#P}$**

- **Fourier Sampling and the power of QTMs**

  In this section we give evidence that QTMs are more powerful than bounded error probabilistic TMs. We define the recursive Fourier sampling problem which on input the program for a boolean function takes on value 0 or 1. We show that the recursive Fourier sampling problem is in BQP.

Basic properties of BQP



Shor

“Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer”

1997年

https://goo.gl/kf4ScC

Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time of at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

1994年、ショアは驚くべき発見をする。（論文は、1997年）

量子コンピュータでは、古典的なコンピュータでは指数関数的時間を要する素因数分解が、多項式時間で解けるという発見である。

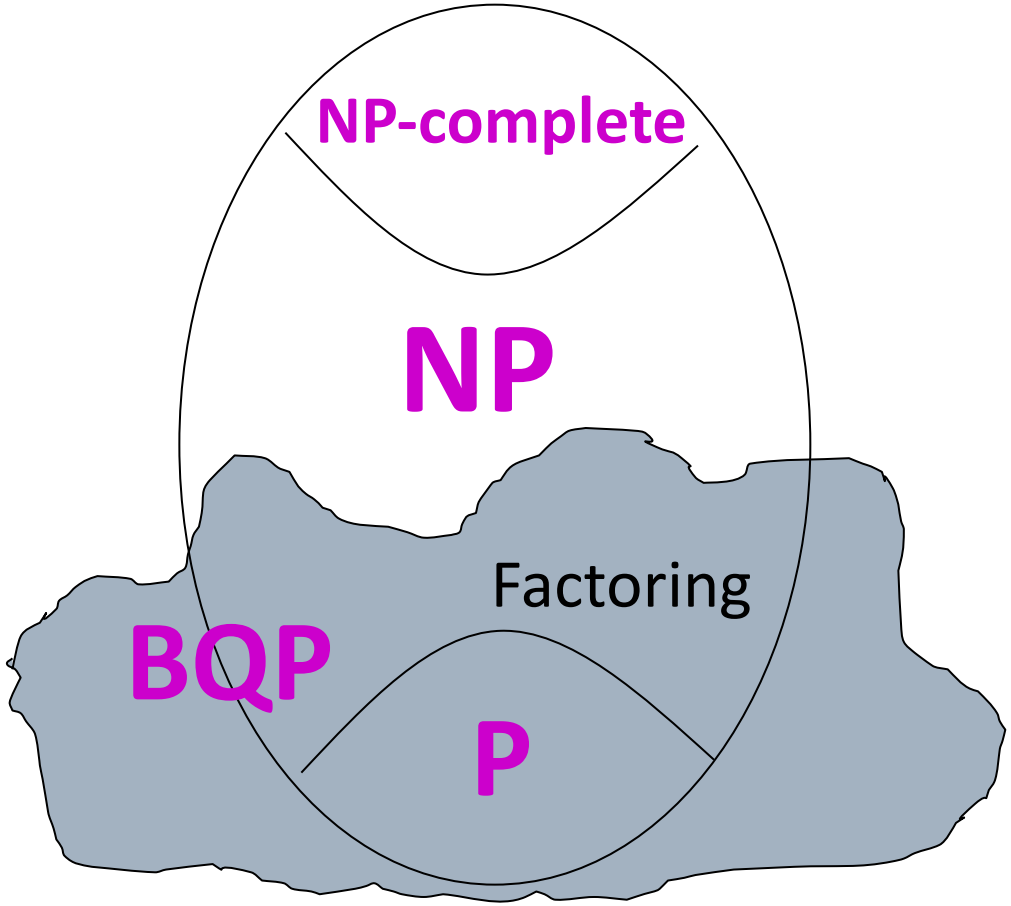
現代のセキュリティ技術の根幹部分が、RSA暗号のように素因数分解の困難性に基礎を置いている事情から、この「ショアのアルゴリズム」の発見は、コンピュータ・サイエンスの枠を超えた、社会的と言っていい大きな反響を呼び起こした。

If computers that you build are quantum,
Then spies everywhere will all want 'em.
Our codes will all fail,
And they'll read our email,
Till we get crypto that's quantum, and daunt 'em.

BQP (Bounded-Error Quantum Polynomial-Time): The class of problems solvable efficiently by a quantum computer, defined by Bernstein and Vazirani in 1993



Shor 1994: Factoring integers is in **BQP**







Extended Church-Turing Thesis

“P ?= NP”

Scott Aaronson

2011年

https://goo.gl/bGwYJG

Extended Church-Turing Thesis

The time it takes to compute something on any one machine is polynomial in the time it takes on any other machine

Even before quantum computing, there were indications that the Extended Church-Turing Thesis might be on shakier ground than the original Church-Turing Thesis. For example, randomized algorithms are sometimes faster than deterministic algorithms, leading to a class called BPP(Bounded-Error Probabilistic Polynomial-Time). Obviously $P \subseteq BPP$. (Today we believe that BPP =P,but we can't prove it.)

https://goo.gl/b2Z1M9

More precisely, the Church-Turing Thesis holds that virtually any model of computation one can define will be equivalent to Turing machines, in the sense that Turing machines can simulate that model and vice versa. A modern refinement, the Extended Church-Turing Thesis, says that moreover, these simulations will incur at most a polynomial overhead in time and memory. Nowadays, most computer scientists and physicists conjecture that quantum computation provides a counterexample to the Extended Church-Turing Thesis—possibly the only counterexample that can be physically realized.

It’s also conceivable that access to a true random-number generator would let us violate the Extended Church-Turing Thesis, although most computer scientists conjecture that it doesn’t, for reasons that I’ll explain in Section 5.4.1. On the other hand, as long as we’re talking only about classical, digital, deterministic computation, the Extended Church-Turing Thesis remains on extremely solid ground.

Conjecture P = BPP

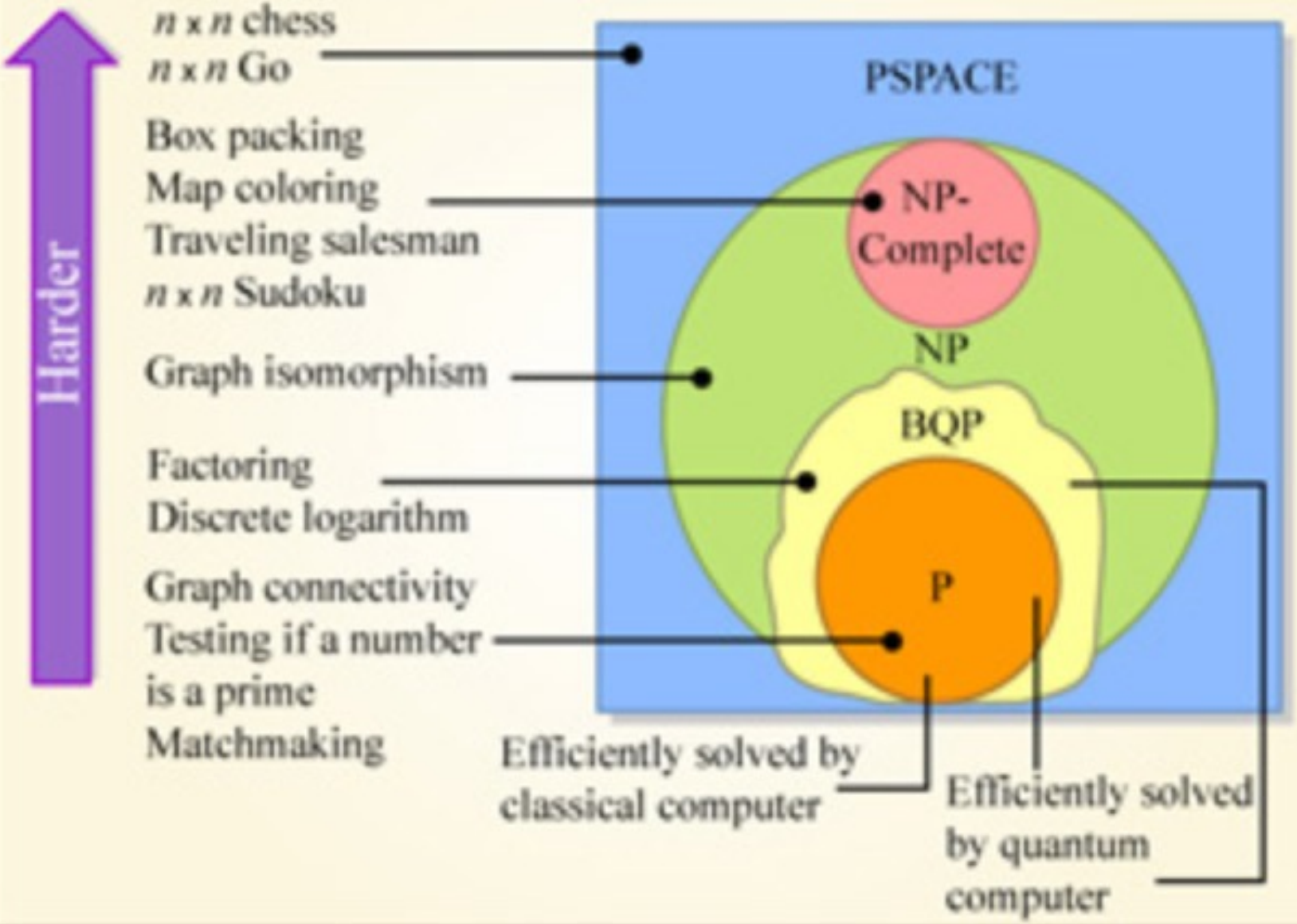
Of course, if P = BPP, then the question of whether randomized algorithms can efficiently solve NP-complete problems is just the original P ?= NP question in a different guise. Ironically, however, the "obvious" approach to proving P = BPP is to prove a strong circuit lower bound—and if we knew how to do that, perhaps we could prove P ≠ NP as well!

Conjecture $NP \not\subseteq BQP$

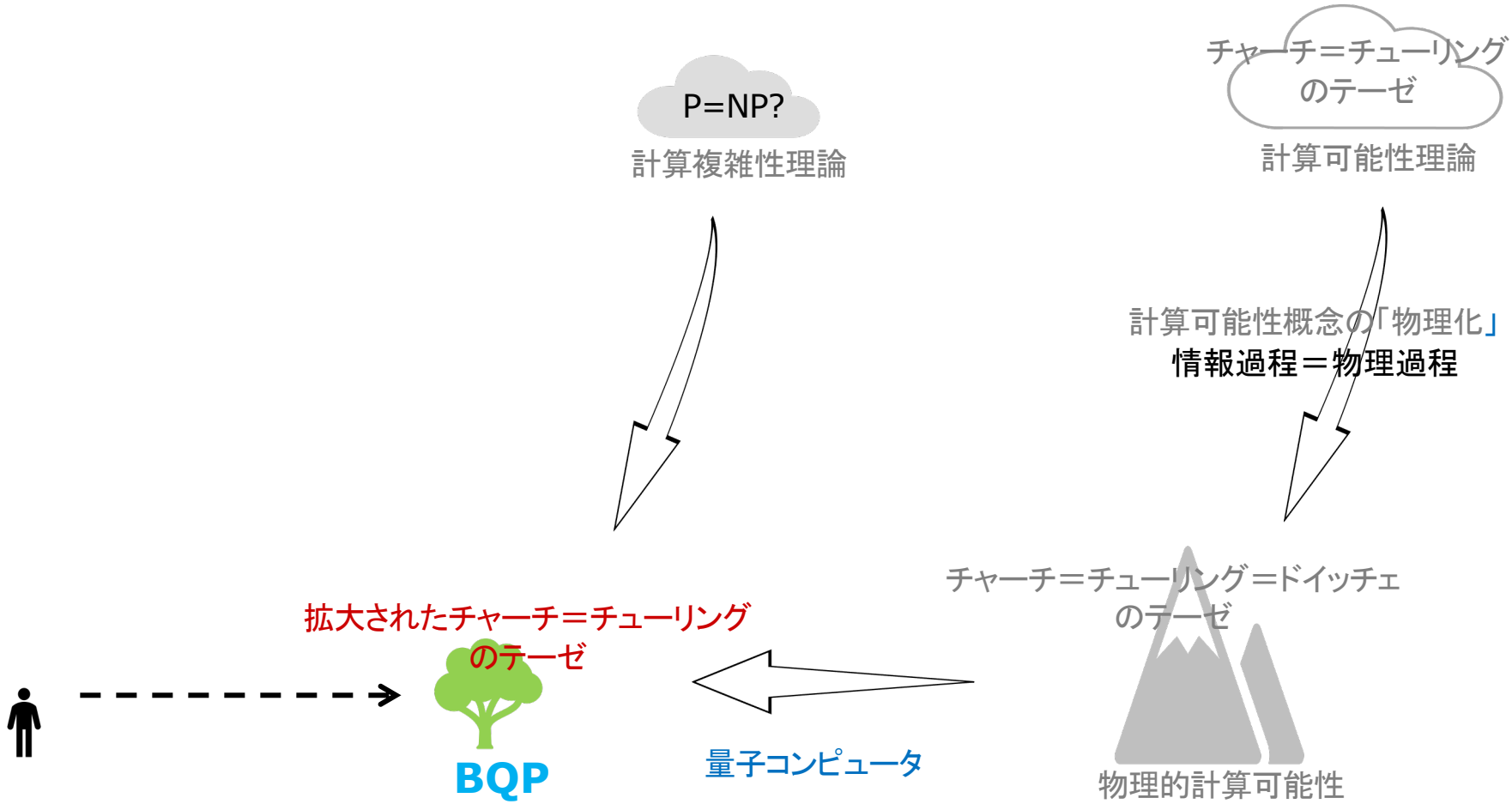
Naturally, there's little hope of proving Conjecture 34 at present, since any proof would imply P /= NP! We don't even know today how to prove conditional statements (analogous to what we have for BPP and P/poly): for example, that if $NP \subseteq BQP$ then PH collapses.

On the other hand, it is known that, if a fast quantum algorithm for NP-complete problems exists, then in some sense it will have to be extremely different from Shor's or any other known quantum algorithm.

Example Problems



90年代



量子複雑性の理論が登場し、BQPという新しい複雑性の概念が確立する。ショアのアルゴリズムは、広い関心をあつめた。

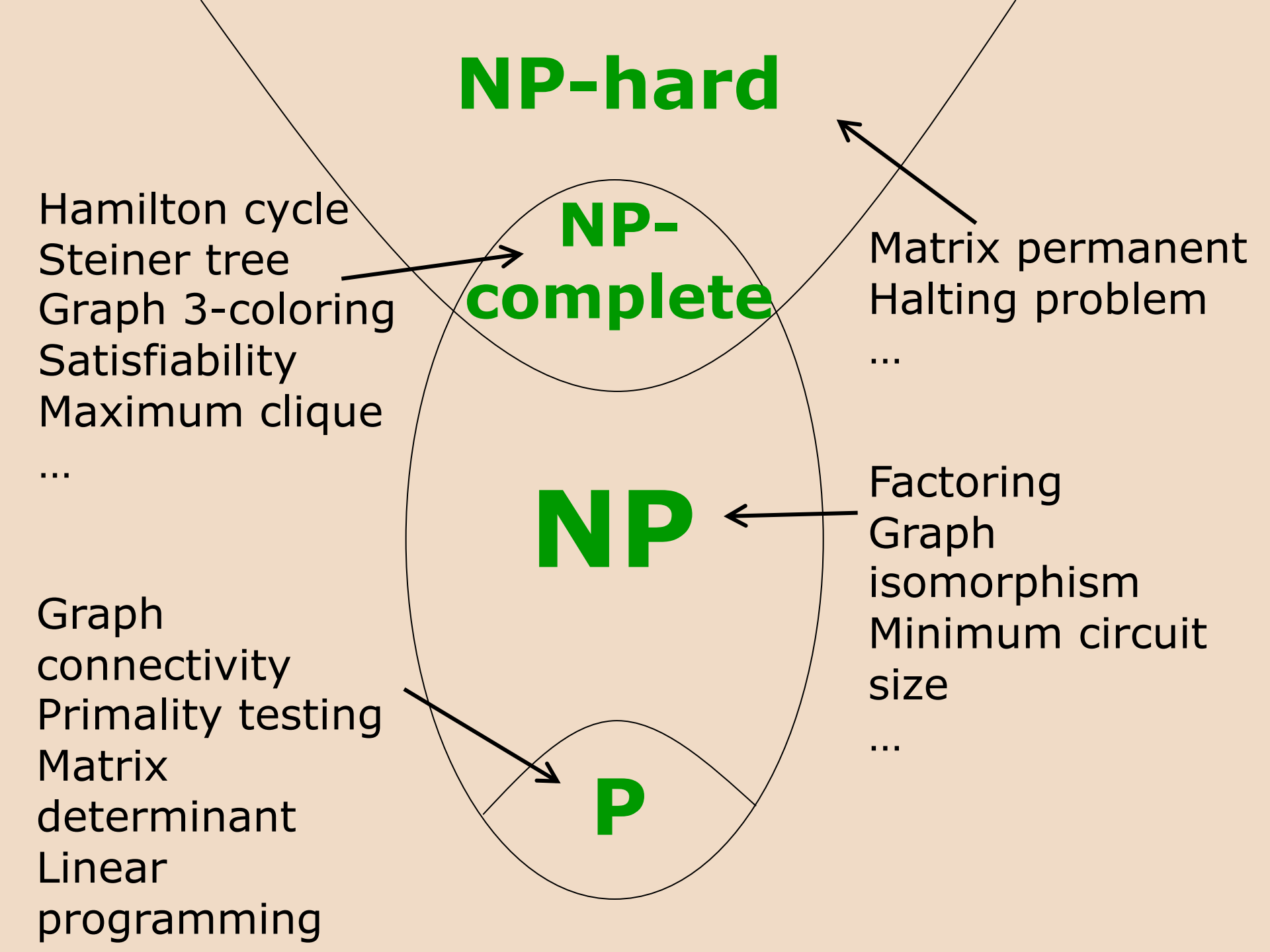
NP-complete Problems and Physical Reality

Scott Aaronson
2005年
https://goo.gl/xyK2Rs

“The limit of Quantum Computer”
Scientific American **298**, 62 - 69 (2008)

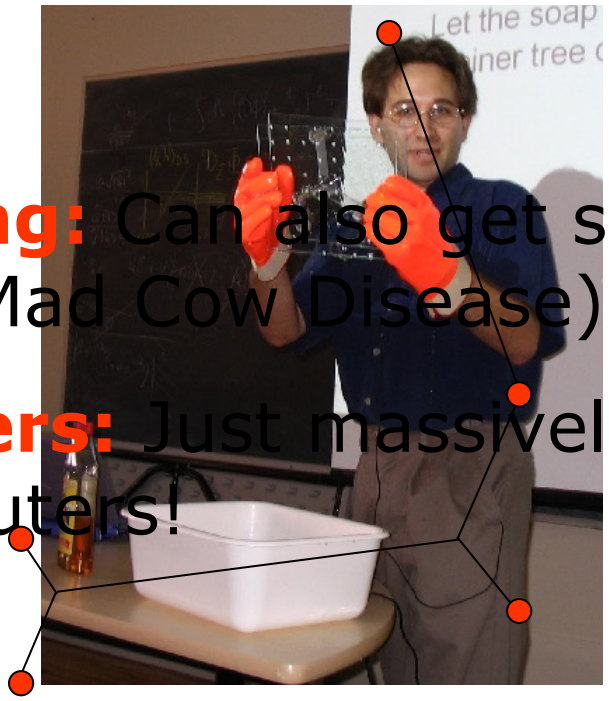
Can NP -complete problems be solved efficiently in the physical universe? I survey proposals including soap bubbles, protein folding, quantum computing, quantum advice, quantum adiabatic algorithms, quantum-mechanical nonlinearities, hidden variables, relativistic time dilation, analog computing, Malament-Hogarth spacetimes, quantum gravity, closed timelike curves, and "anthropic computing." The section on soap bubbles even includes some "experimental" results.

While I do not believe that any of the proposals will let us solve NP -complete problems efficiently, I argue that by studying them, we can learn something not only about computation but also about physics.


NP-hard
NP-complete
NP
P
Hamilton cycle
Steiner tree
Graph 3-coloring
Satisfiability
Maximum clique
…
Matrix permanent
Halting problem
…
Factoring
Graph isomorphism
Minimum circuit size
…
Graph connectivity
Primality testing
Matrix determinant
Linear programming

Other Alleged Ways to Solve NP-complete Problems

Dip two glass plates with pegs between them into soapy water; let the soap bubbles form a minimum "Steiner tree" connecting the pegs (thereby solving a known **NP**-complete problem)



Protein folding: Can also get stuck at local optima (e.g., Mad Cow Disease)

DNA computers: Just massively parallel classical computers!

The NP Hardness Assumption

There is no physical means to solve NP complete problems in polynomial time.

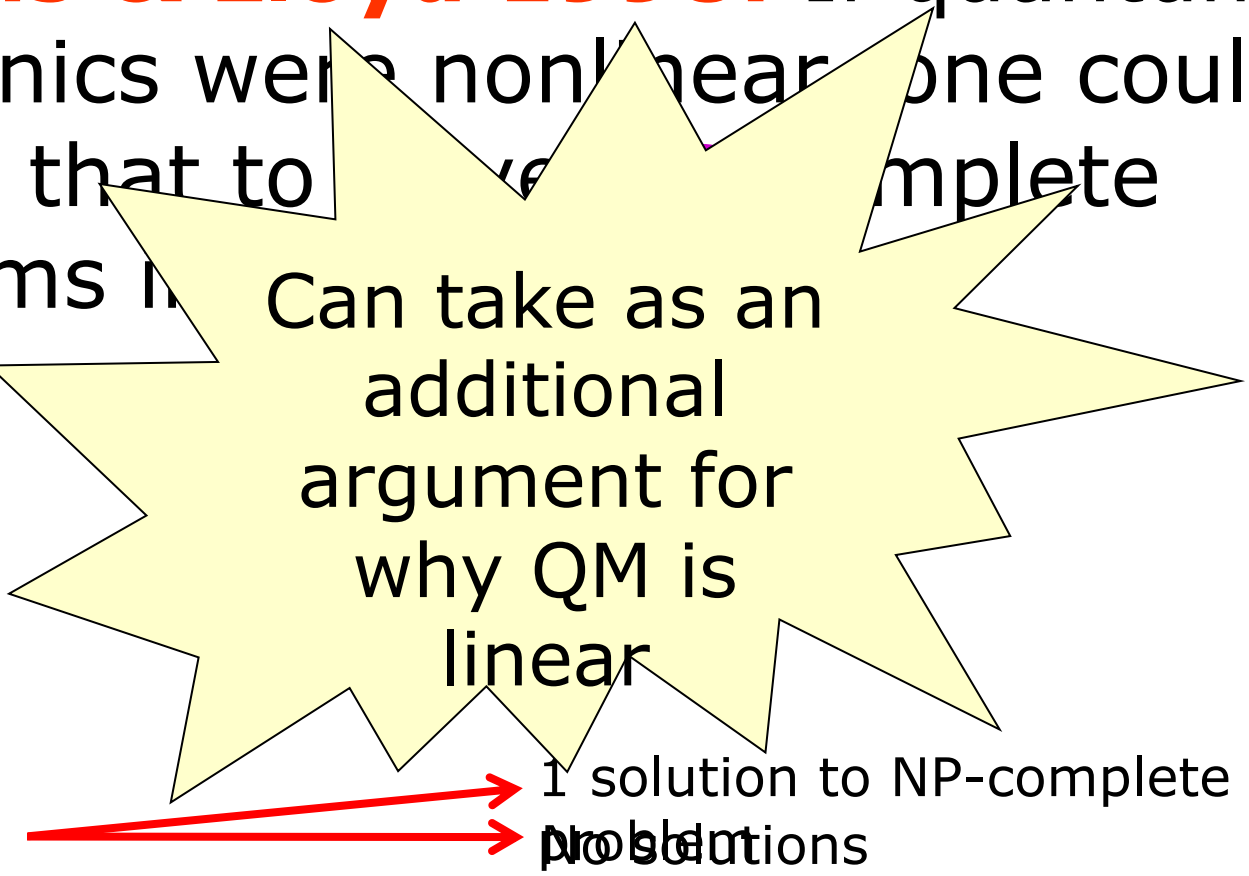
Rest of talk: Show how complexity yields a new perspective on linearity of QM, anthropic postselection, closed timelike curves, and initial conditions

Alright, what c
assumption?

- Implies, bu
- As falsifiab
- Consistent with currently known physical theory
- **Scientifically fruitful?**

1. Nonlinear variants of the Schrödinger Equation

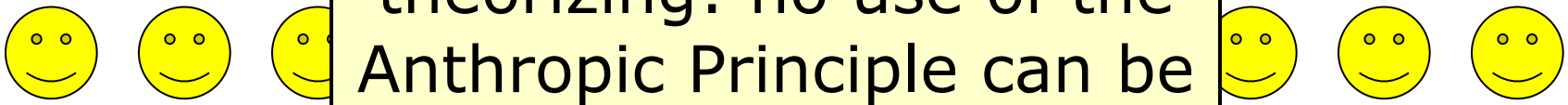
Abrams & Lloyd 1998: If quantum mechanics were nonlinear, one could exploit that to [illegible] NP-complete problems in



2. Anthropic Principle

Foolproof way to solve **NP**-complete problems in polynomial time (at least in the Many-Worlds Interpretation)

First gues…en, if it's wrong, ki…



Technica…tions, you're ou…

Solution: With tiny probability don't do anything. Then, if you find yourself in a universe where you didn't do anything, there probably *were* no solutions, since otherwise you would've found one!

NP Hardness Assumption yields a nontrivial constraint on anthropic theorizing: no use of the Anthropic Principle can be valid, if its validity would give us a way to solve **NP**-complete problems in polynomial time

What if we combine quantum computing with the Anthropic Principle?

I.e. perform a polynomial-time quantum computation, but where we can measure a qubit and **assume** the outcome will be $|1\rangle$



Leads to a new complexity class: **PostBQP** (Postselected **BQP**)



Certainly **PostBQP** contains **NP**—but is it even bigger than that?

3. Time Travel

Everyone's first idea for a time travel computer: Do an arbitrarily long computation, then send the answer back in time to before you started

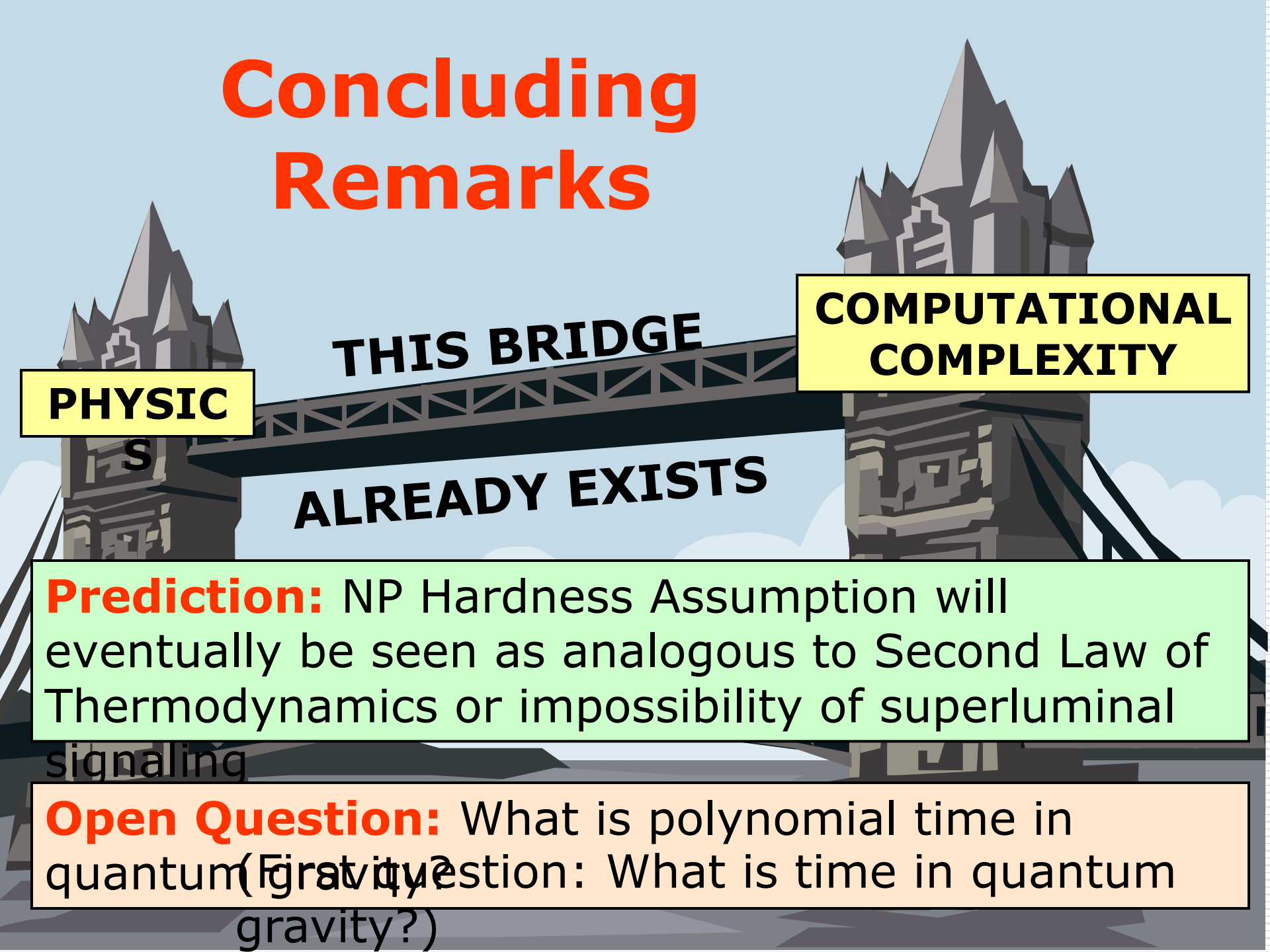
THIS DOES NOT WORK

Why not?



- Ignores the Grandfather Parad
- Doesn't take into account the computation you'll have to do **after** getting the answer

Concluding Remarks



Prediction: NP Hardness Assumption will eventually be seen as analogous to Second Law of Thermodynamics or impossibility of superluminal signaling

Open Question: What is polynomial time in quantum gravity? (Firstly question: What is time in quantum gravity?)

物理学と量子情報理論


I
COMPLEXITY

ブラックホールのエントロピーの発見

“Black holes and Entropy”
Jacob D. Beckenstein
http://www.phys.huji.ac.il/~bekenste/PRD7-2333-1973.pdf
1973年

ブラックホール「No Hair 定理」

- ブラックホールは、観測可能な物理量としては、質量・電荷・角運動量の三つしか持たないと信じられてきた。ブラックホールは、極めて単純な構造の存在で、上の三つの物理量でその特徴を、完全に記述できる。それを、物理学者のWheelerは、「ブラックホールには毛がない」といった。
- ブラックホールには、マイクロステーツがない？
- ベッケンシュタインの発見は、これを覆すもの。彼は、ブラックホールのエントロピー S_{BL} が、その地平の表面積 A に比例することを発見する。比例定数は、のちにホーキングが1/4であることを見つける。

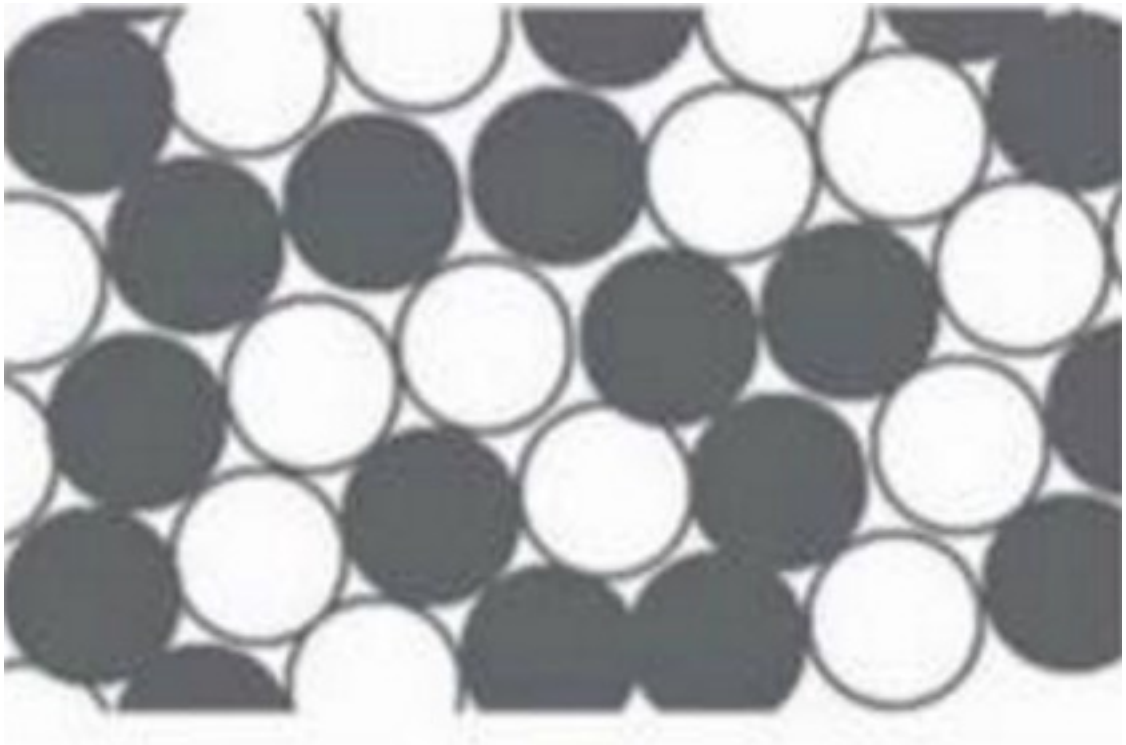
$$S_{BL} = \frac{1}{4} A$$

サスキンドの計算

- 今、ブラックホールに、フォトン（光の粒子）が一個落ち込んだとする。このフォトンは、1bitの情報を持っているとする。
（本当は、フォトン一個は、無数の情報を持ちうるのだが。フォトンの波長を、ブラックホールの地平の半径と同じ程度に広いものにすれば、フォトンの輪郭はぼやけ、それが持つ情報を小さなものにできる）
- この時、ブラックホールのエネルギーは、どれほど増えるだろうか？　それは、ブラックホールに落ち込んだ、1bitの情報を運ぶフォトンのエネルギーだけ増える。その量は計算できる。
- エネルギーは質量に等しいので、この時、ブラックホールの質量が、どれだけ増えるかがわかる。
- ブラックホールの地平の半径Rは、ブラックホールの質量Mで決まる。半径がわかれば、その地平の面積の変化を計算できる。

- 1bitの情報を持つフォトン一個が、ブラックホールに落ち込むと、ブラックホールの質量は、10^{-45}キログラム増えるという。小数点のゼロの下にゼロが45個並ぶ数だから、普通なら、無視しても構わない変化だ。
- この時、地平の半径は、10^{-72}メーター伸びるという。さっきの変化より、もっともっと小さい。普通なら、無視しても構わない変化だ。
- この時、地平の面積は、10^{-70}メーター広くなる。これも、無視して構わない変化だ。
- サスキンドは、ここであることに気づく。10^{-70}というのは、プランクの長さ h= 10^{-35}の二乗じゃないか？
- そこで、ブラックホールの質量を、いろいろ変えて計算すると、驚くべきことに、どんな質量のブラックホールでも、1bitの情報を持つフォトンが飛び込むと、その地平の面積は h^2 プランク長の二乗分だけ広がるのである！（一部、脚色あり。）

- イメージとしては、こういうこと。
- ブラックホールの地平が、ｈｘｈ のマス目で、びっしり埋められているとする。ここに、1bitの情報をもったフォトンが、飛び込むと、新しいマス目が、他を押しのけて一つ増えるということ。なんか、格好いい発見である。



新しい情報 / エネルギー観

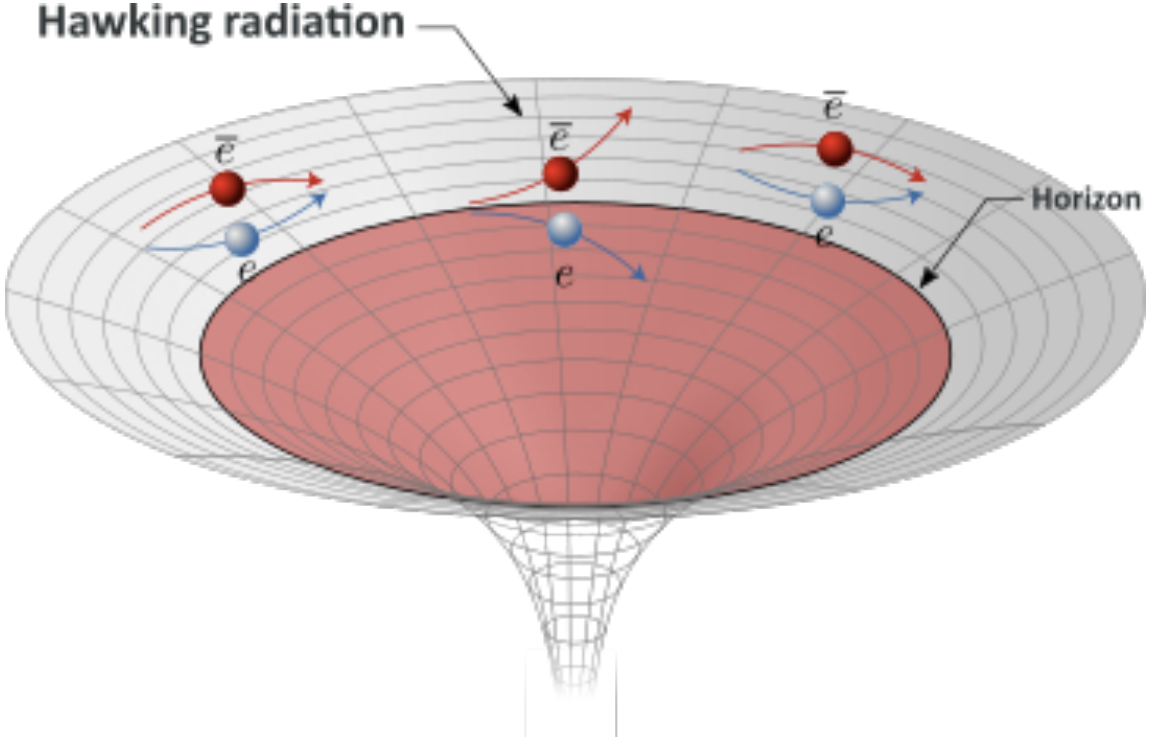
- 落ち込んだフォトンの情報は、地平の平面にへばりつくのだ。
- これは、「ブラックホールのエントロピーは、地平の面積に比例する。」という、ベッケンシュタインの重要な発見を説明し、さらに詳しく言い換えたものだ。
- 「ブラックホールのエントロピーは、bitで計測すれば、プランクの面積単位で計測した地平の面積に比例する。」
- 「温度とは、1bitの情報が追加された時の、システムのエネルギーの上昇である。」
- これは、19世紀のボルツマン、20世紀のシャノンを超える、おそらく、21世紀の科学を特徴付ける、新しい 情報 / エネルギー観である。

ブラックホール輻射

S. W. Hawking
“Black hole explosion?”
https://www.nature.com/nature/journal/v248/n5443/abs/248030a0.html
1974年

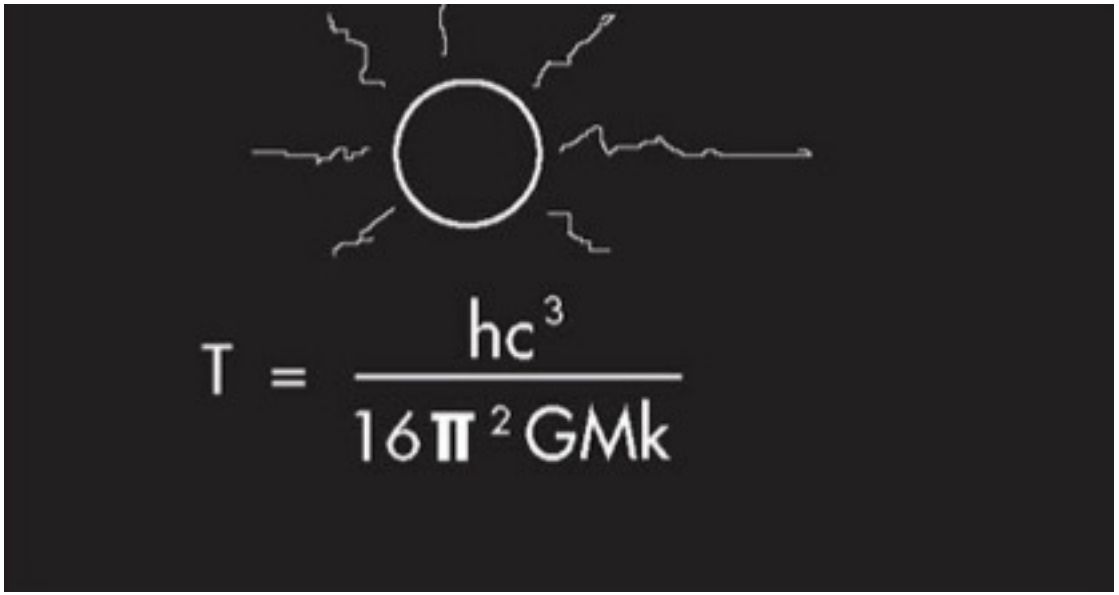
ブラックホール輻射とブラックホールの蒸発

- ブラックホールの地平周辺で対発生した粒子の一方がブラックホールに落ち込み、た方がそこから抜け出すと、ブラックホールから粒子が飛び出すように見える。ブラックホールは、それによってエネルギーを失って、じょじょに蒸発していく。

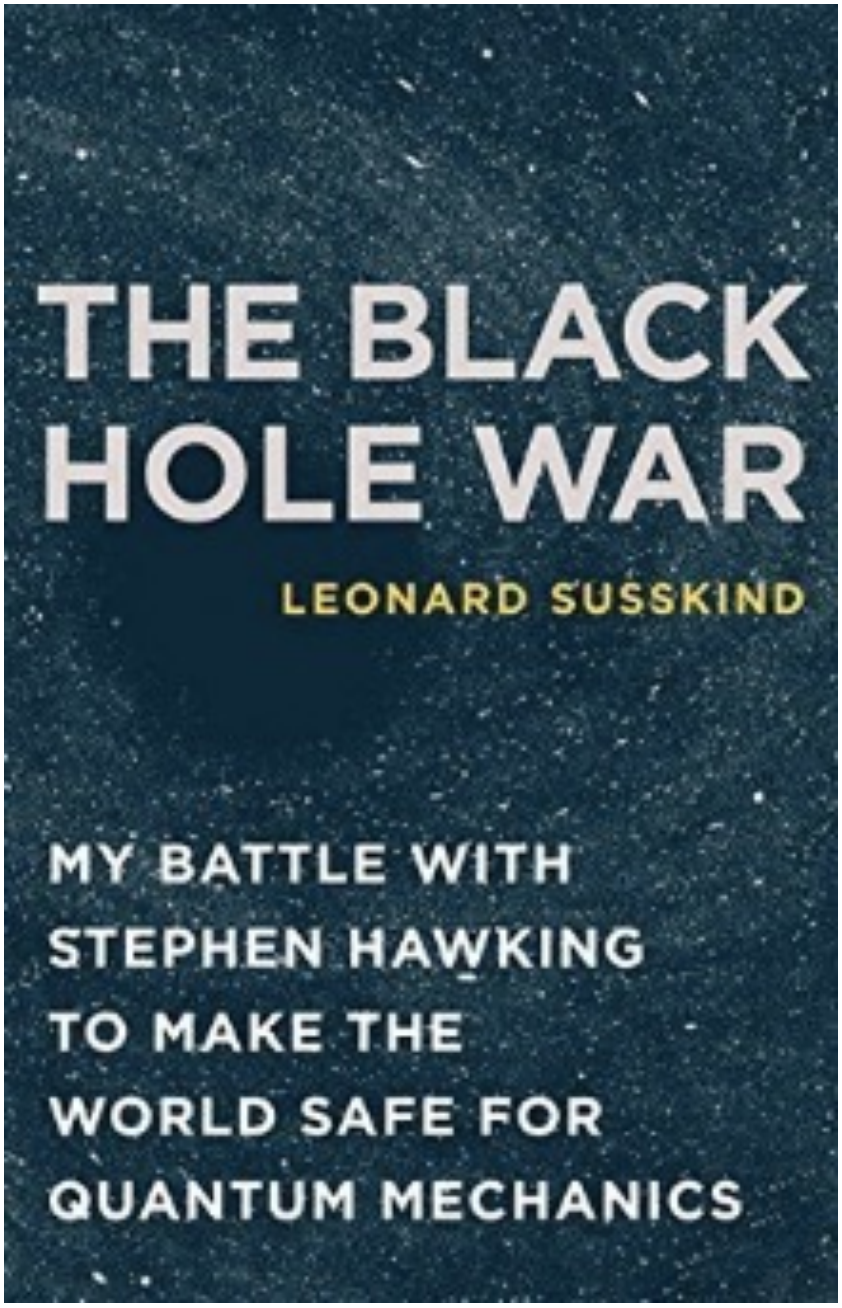


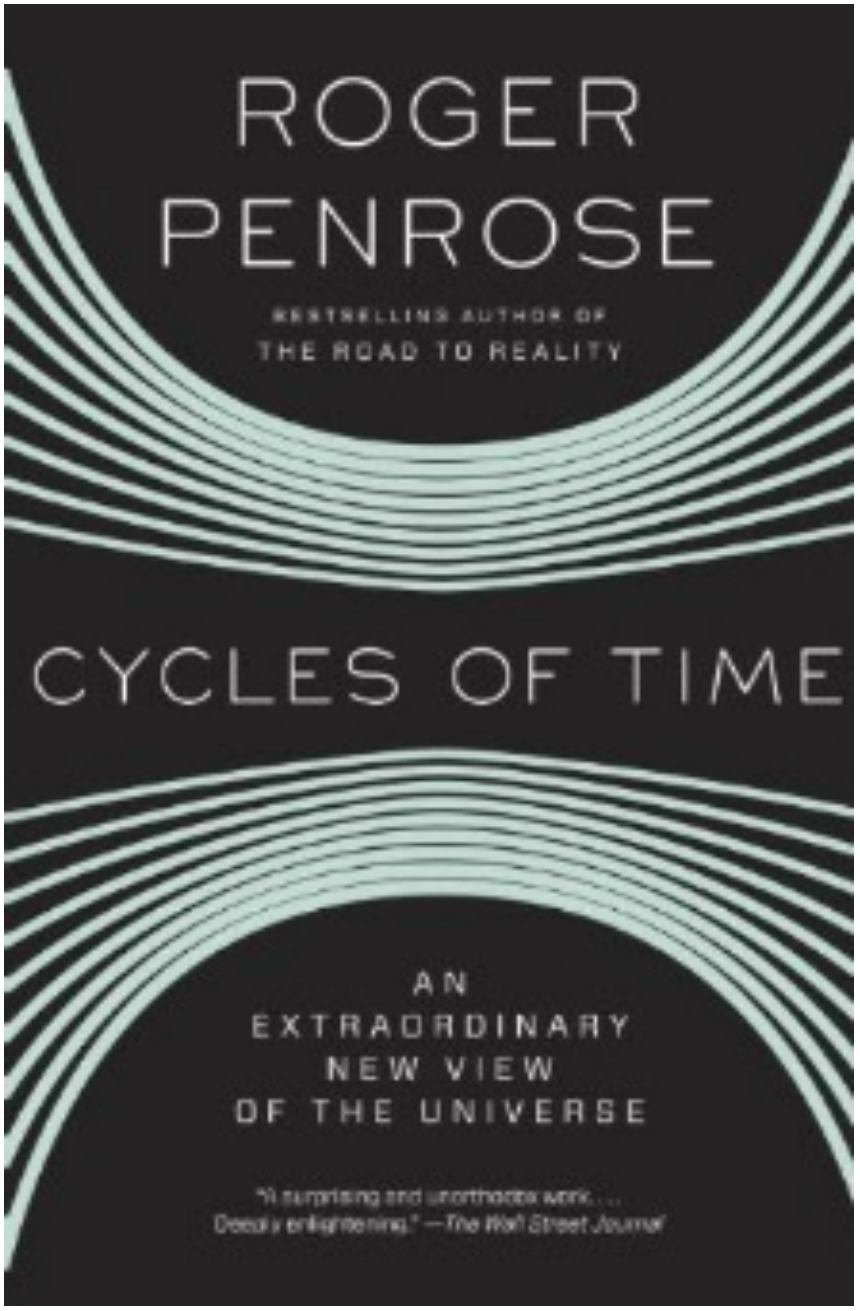
ブラックホールは、「温度」も持つ

- この複写により、ブラックホールは、温度も持つことになる。ホーキングは、その温度も導出した。ブラックホールは質量 *M* が小さければ小さいほど高温であるといえる。とはいえその温度は、例えば太陽の数倍の質量を持つブラックホールの場合、100万分の1 K 程度となり、通常の恒星質量クラスのブラックホールでは宇宙背景放射の温度（3 K）よりもずっと低い。



ブラックホールで情報は消滅するか


THE BLACK HOLE WAR
LEONARD SUSSKIND
MY BATTLE WITH STEPHEN HAWKING TO MAKE THE WORLD SAFE FOR QUANTUM MECHANICS


ROGER PENROSE
BESTSELLING AUTHOR OF THE ROAD TO REALITY
CYCLES OF TIME
AN EXTRAORDINARY NEW VIEW OF THE UNIVERSE

AdS/CFT 対応

J. M. Maldacena,

“The large N limit of superconformal field theories and supergravity”

https://arxiv.org/pdf/hep-th/9711200.pdf

1998年1月

量子論と相対論の統一の課題

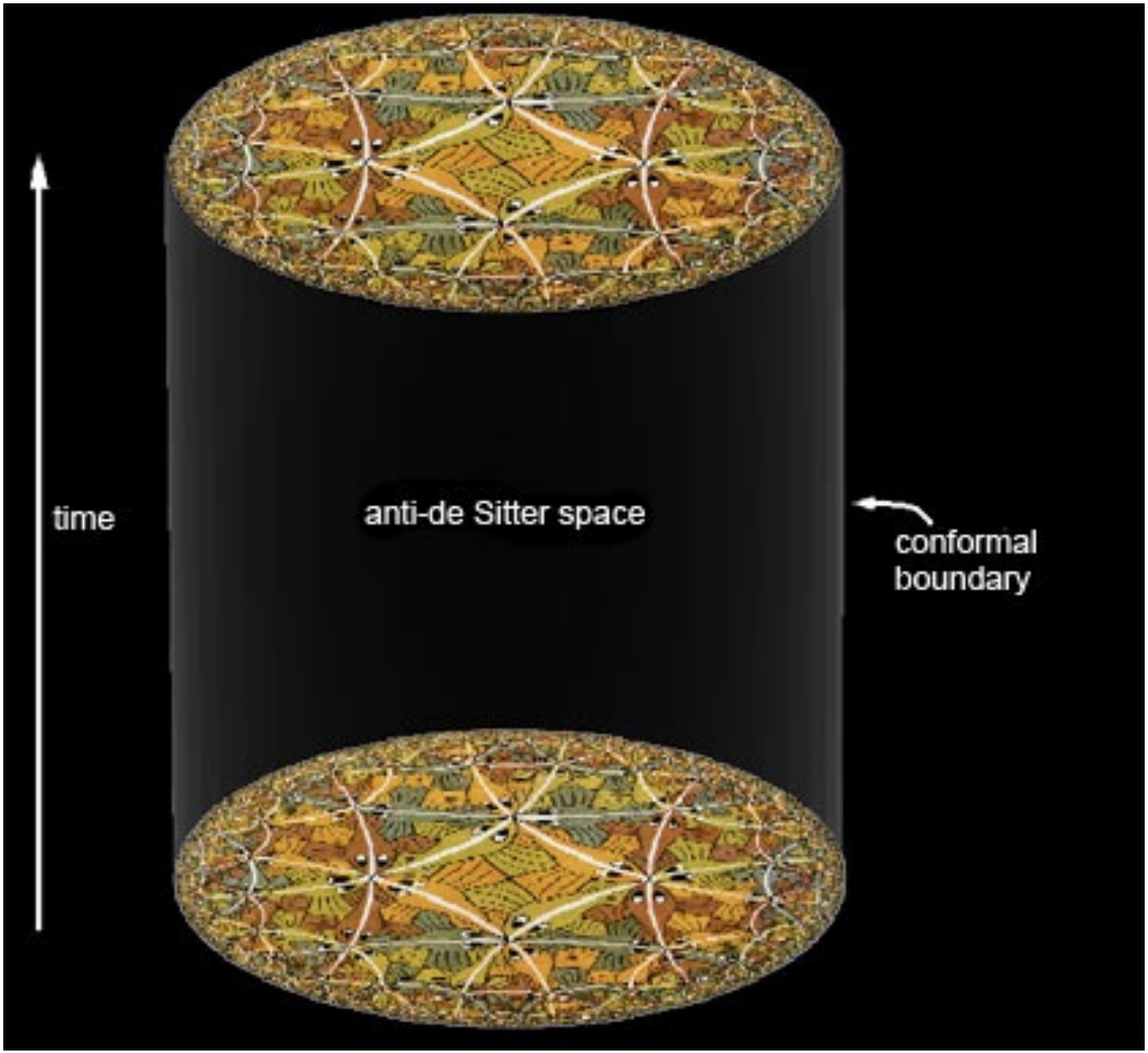
- 極微な世界を記述する量子力学と巨大な時空を記述する重力の理論である相対論との統一は、物理学の難問である。
  二つの理論は、それぞれの領域では、100年近くの間、大きな成功をおさめてきたのだが。それぞれ異なる理論体系に基づく、物理学が二つある？ それは、奇妙なことだ。
- このところ、この問題が解決できるのではという希望が生まれているように見える。もっとも重要な飛躍は、物理学者マルデセーナによる次のような発見である。（だいぶ、テキトーにまとめている）
- 「キャンベルのスープ缶にたとえれば、重力理論はスープの理論で、量子論は缶の理論である。」
- ブリキの缶の外側をいくらなぞっても、スープのことはわからない。逆に、いくらスープを舐めても、缶のことはわからない。

量子論と相対論の思わぬ接点

- マルデセーナがエライのは、そうではないことを見つけたことである。「スープと缶は、無関係ではなく関係がある。缶を調べれば、スープのことがわかり、スープを調べれば缶のことがわかる！」

- それは、文字通り「接点」だった。重力理論（スープ）に「境界」を接して量子論（缶）が、棲んでいたのだから。これは、「境界」の重要性の発見でもあった。





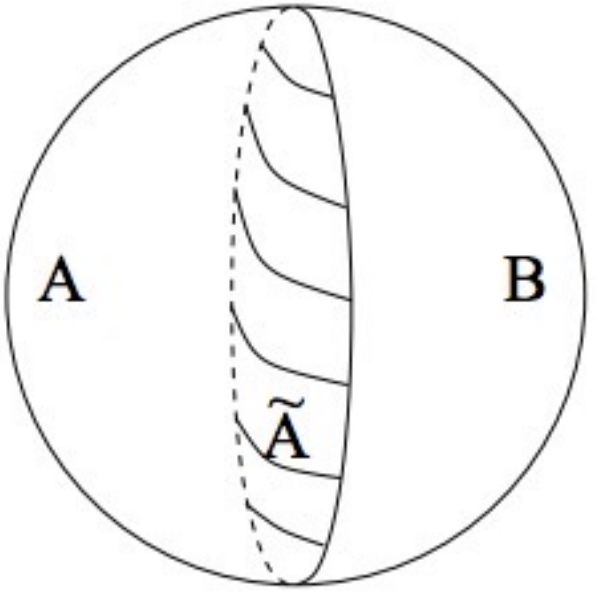
https://goo.gl/imCz3r

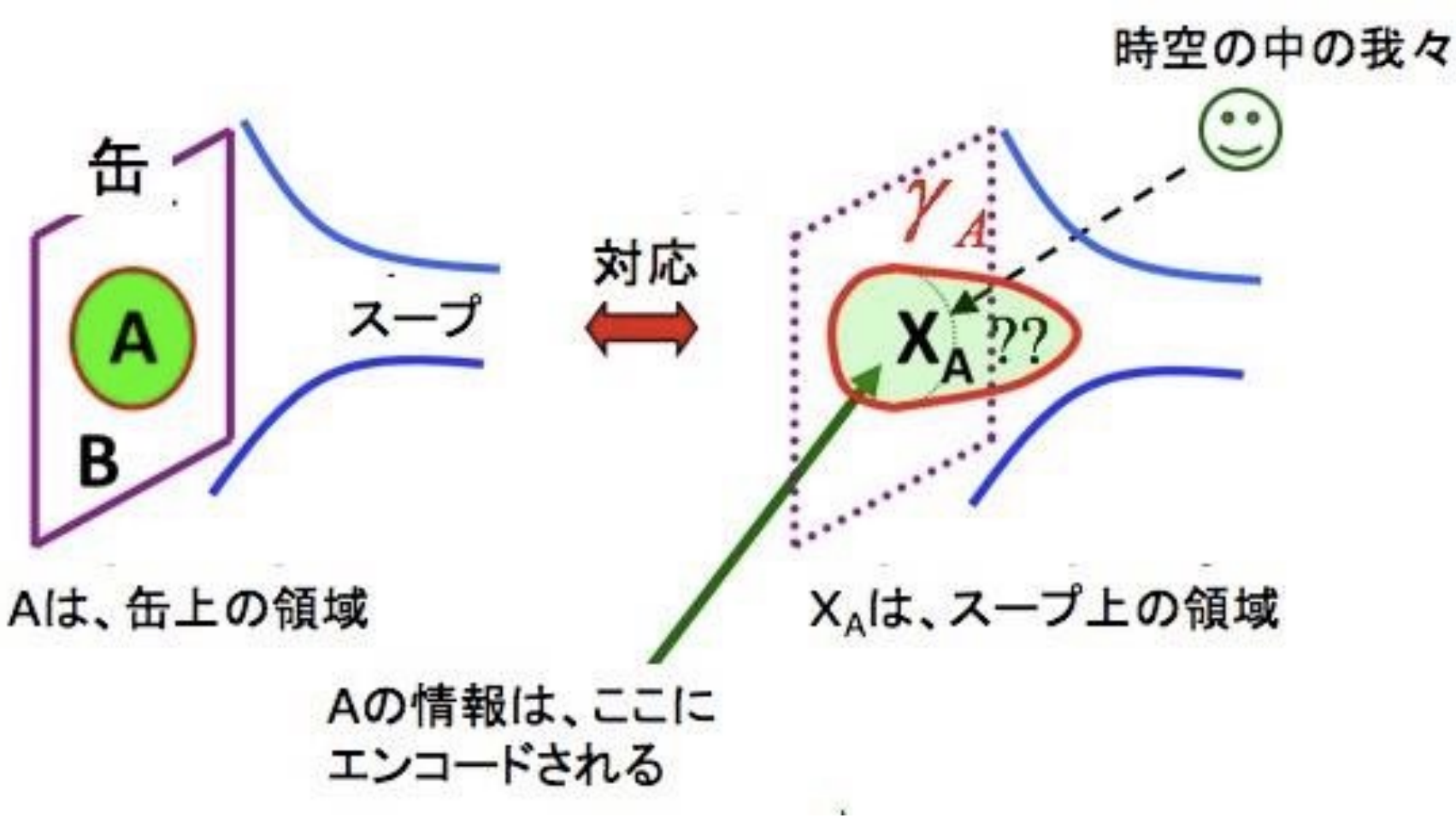
エンタングルメントのエントロピー

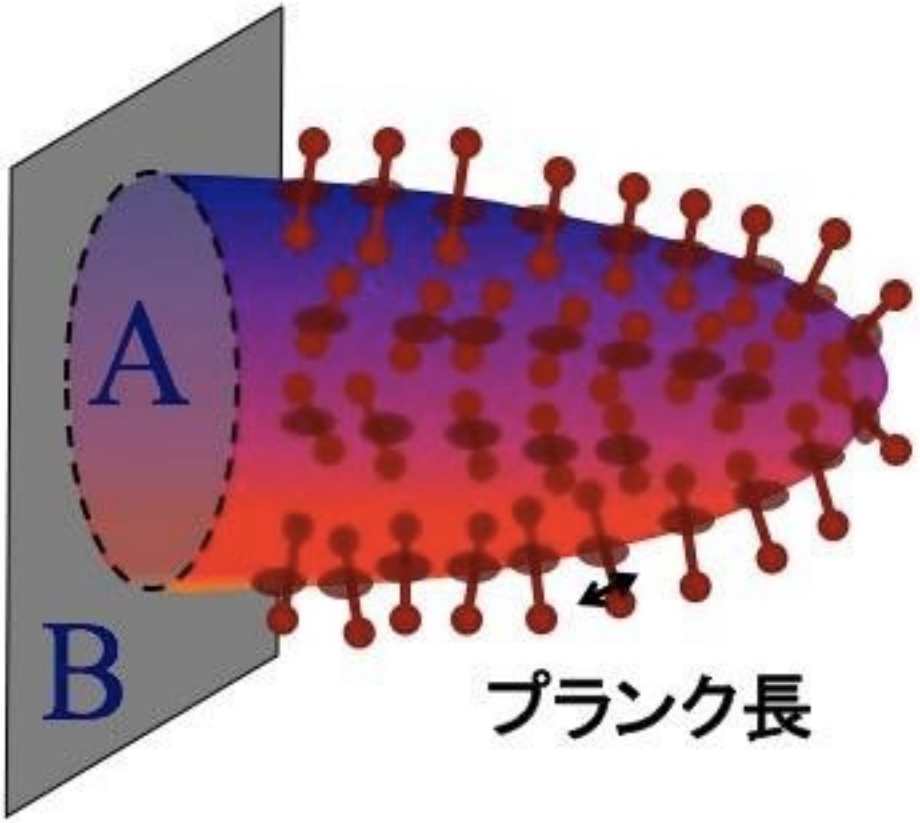
Ryu and Takayanagi,
“Holographic derivation of entanglement entropy from AdS/CFT”
https://arxiv.org/pdf/hep-th/0603001.pdf
2006年5月

日本人物理学者　笠と高柳

- 次に、日本の二人の物理学者、笠と高柳が登場する。
- 「スープ（「時空」だと思っていい）が二つの部分 AとBに別れているとする。AとBの「境界部分」は、スープの「境界」だから、マルデセーナの理論にしたがって量子論で記述できるはず。」（
- 「やってみたら、この「境界」は、なんと、量子論の不思議な「もつれあい（エンタングルメント）」のエントロピーに対応するんだ！」
- 彼らは、「量子もつれあい」がエントロピーを持つことの、最初の発見者だ。これは、ベッケンシュタインによる、ブラックホールがエントロピーを持つことの発見に比肩できる発見だ。




時空の中の我々
缶
A
B
スープ
対応
γ_A
X_A
??
Aは、缶上の領域
X_Aは、スープ上の領域
Aの情報は、ここに
エンコードされる



$$S_A = \frac{\mathrm{Area}(\gamma_A)}{4G_N} \approx \frac{\mathrm{Area}(\gamma_A)}{l_{pl}^2}.$$

時空を結びつけるエンタングルメント

M. Van Raamsdonk,

“Building up spacetime with quantum entanglement”

https://arxiv.org/pdf/1005.3035.pdf

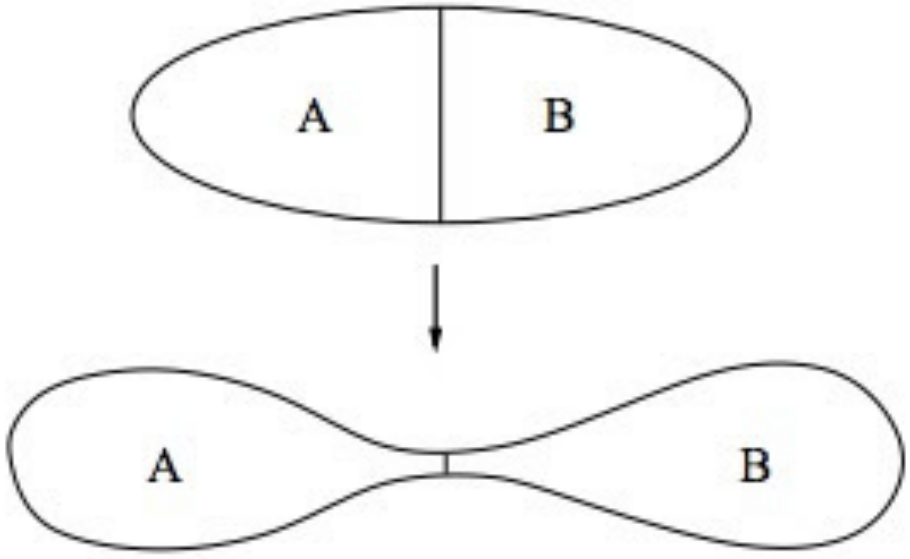
2010年5月

エンタングルメントのエントロピーが時空を縫い合わせる！

- これに刺激されて、今度は、カナダのラームズダンクがくる。（以下も、テキトー。もっとも、彼の最初の頃の論文は、テキトーなこと言っているとみなされて、どの学会誌からも、掲載を拒否されたらしい。）
- 「二つの時空 A, B があったとする。二つの時空を、引き離してみる。そうすると、「境界面」の面積は減少する。これは、二つの時空が離れれば離れるほど、もつれあいのエントロピーが減ることを意味している。このエントロピーがゼロになった時、二つの時空は、引きちぎられる。」

エンタングルメントのエントロピーが時空を縫い合わせる！

- 「そうだ！　逆に考えればいいんだ。 時空を結びつけているのは、もつれあいなんだ。エンタングルメントのエントロピーが、時空を縫い合わせているんだ。」
- これは、巨大な飛躍だった。



Tensor Network

Roman Orus

"A Practical Introduction to Tensor Networks: Matrix Product States and Projected Entangled Pair States"

https://arxiv.org/pdf/1306.2164.pdf

凝縮系物性論のチャレンジ

- 液体ヘリウムが、超低温状態で超電導・超流動の性質を示すことは、バーディーン、クーパー、シュリーファーのBCS理論で、電子等のフェルミ粒子が、凝縮してボース粒子の性質を示す（一つの系で、複数の粒子が同一の状態を示す）ことで説明される。凝縮系物性論は、こうした物質の新しい相を研究する。
- 現在でも、超電導技術は、リニア新幹線や医療でのMRIのように広く利用され、また、量子コンピュータでも基本的な要素技術として利用される。凝縮系物性論は、テクノロジーを大きく変える可能性を持つ、チャレンジングな分野である。
- ただ、現在では、高温超電導のような現象を、理論的に説明することはできない。

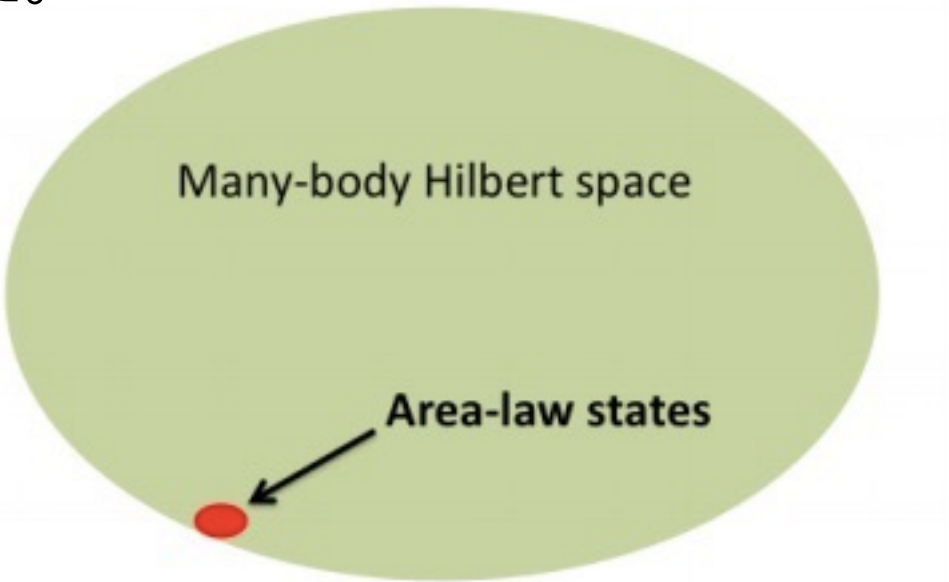
量子多体系の問題を解く難しさ

- 凝縮系物性論のターゲットとする領域を、「量子多体系」と呼ぶことがある。多体系の量子論的状態を明らかにするためには、粒子の数Nに対応して、（状態の数）^N の次のような複雑な方程式をヒルベルト空間上で解く必要がある。
- N=10^{23}　だとしても、その複雑さは、10^(10^{23})にもなる。宇宙の素粒子の数が、10^90程度だというから、これを解くのは事実上、不可能である。

$$\langle\Psi|\hat{H}|\Psi\rangle = \sum_{\sigma_1,\ldots,\sigma_L}\sum_{\sigma'_1,\ldots,\sigma'_L}\sum_{a_1,\ldots,a_L}\sum_{a'_1,\ldots,a'_L}\sum_{b_1,\ldots,b_L}$$
$$A^{*\sigma_1}_{1,a_1} W^{\sigma_1,\sigma'_1}_{1,b_1} A^{\sigma'_1}_{1,a'_1} A^{*\sigma_2}_{a_1,a_2} W^{\sigma_2,\sigma'_2}_{b_1,b_2} A^{\sigma'_2}_{a'_1,a'_2} \cdots$$
$$\cdots A^{*\sigma_{L-1}}_{a_{L-2},a_{L-1}} W^{\sigma_{L-1},\sigma'_{L-1}}_{b_{L-2},b_{L-1}} A^{\sigma'_{L-1}}_{a'_{L-2},a'_{L-1}} A^{*\sigma_L}_{a_{L-1},1} W^{\sigma_L,\sigma'_L}_{b_{L-1},1} A^{\sigma'_L}_{a'_{L-1},1}$$

エントロピーが「Area則」を満たす
基底状態の計算の簡素化が可能に

- 「Area則（面積則）」というのは、エントロピーが系の体積ではなく系の境界の面積に比例するという法則のこと。エンタングルメントのエントロピーは、Area則を満たす。
- この時、量子多体系の基底状態の解は、一般の巨大なヒルベルト空間の中ではなく。ごく限られた領域の中にあることが、明らかになった。



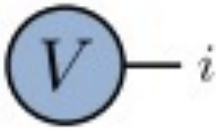
Tensor Networkが、グラフでその解を表現する

Notation

- Tensor Network は、グラフとして表現される

ベクトル

$$V_i = \begin{pmatrix} V_1 \\ V_2 \end{pmatrix}$$



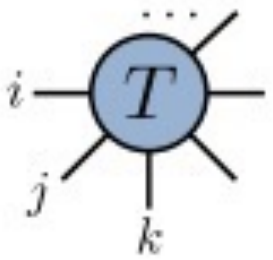
行列 ::

$$M_{ij} = \begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix}$$



- テンソル

$$T_{ijk\ldots}$$



https://goo.gl/Zq9WUw

テンソルの縮約

つなげた線が、縮約を表す

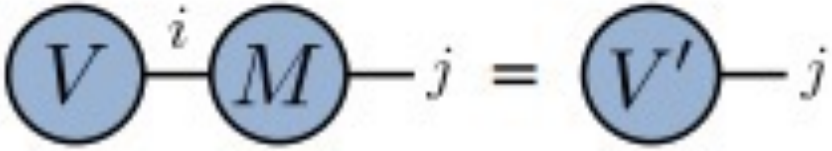
- 二つのベクトル

$$\sum_i V_i V'_i = S$$



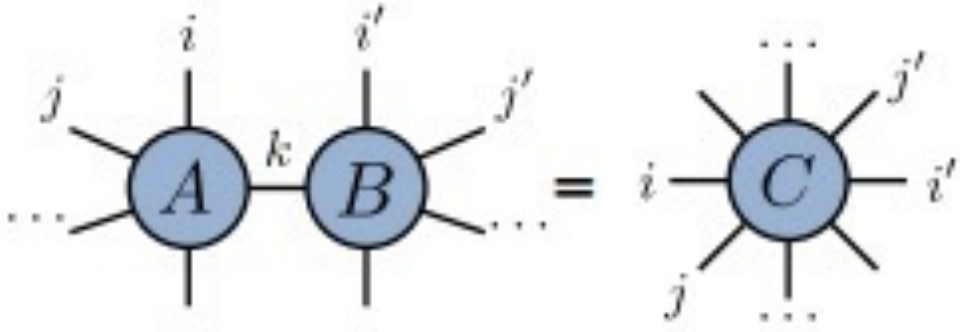
- ベクトルと行列

$$\sum_i V_i M_{ij} = V'_j$$

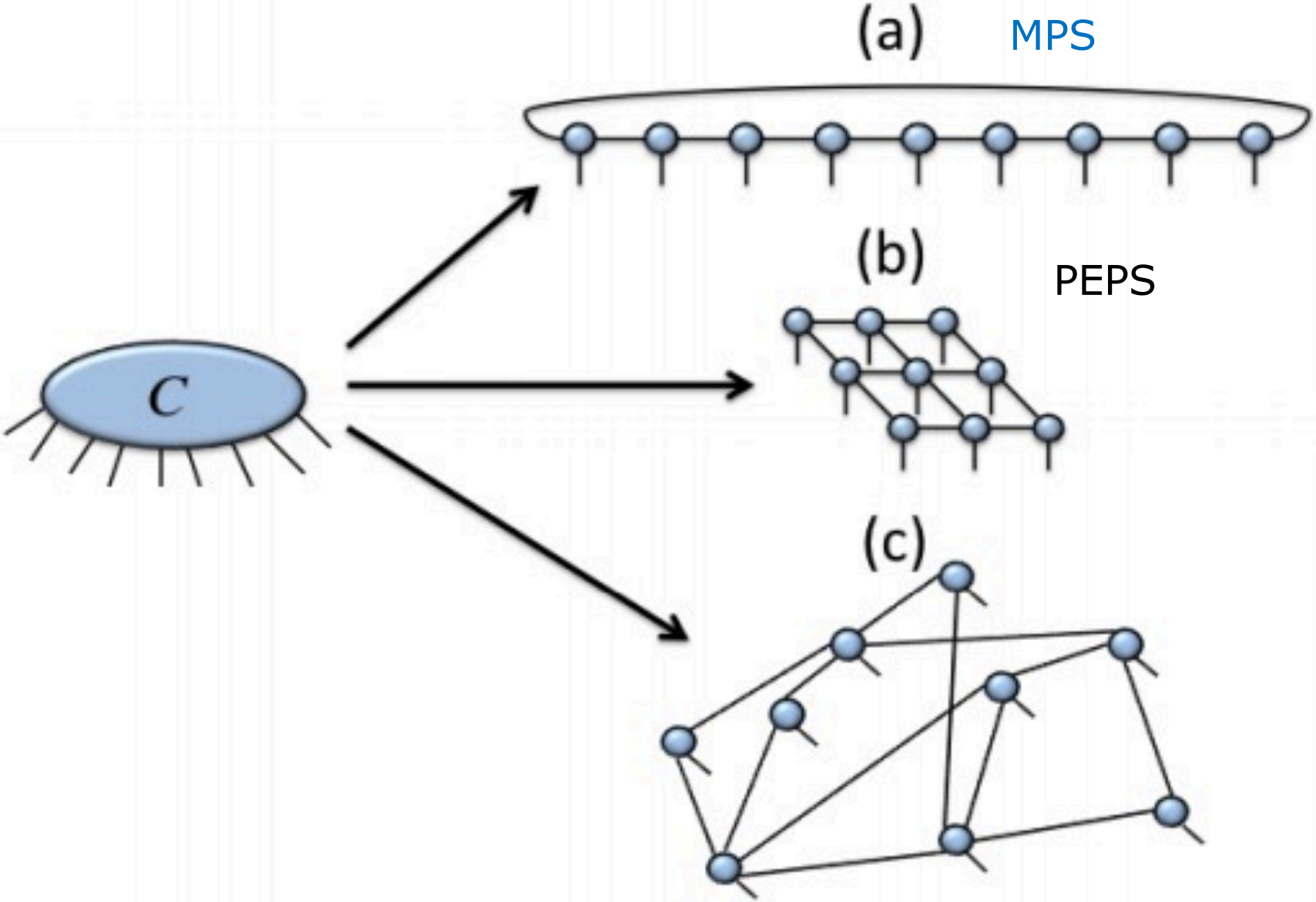


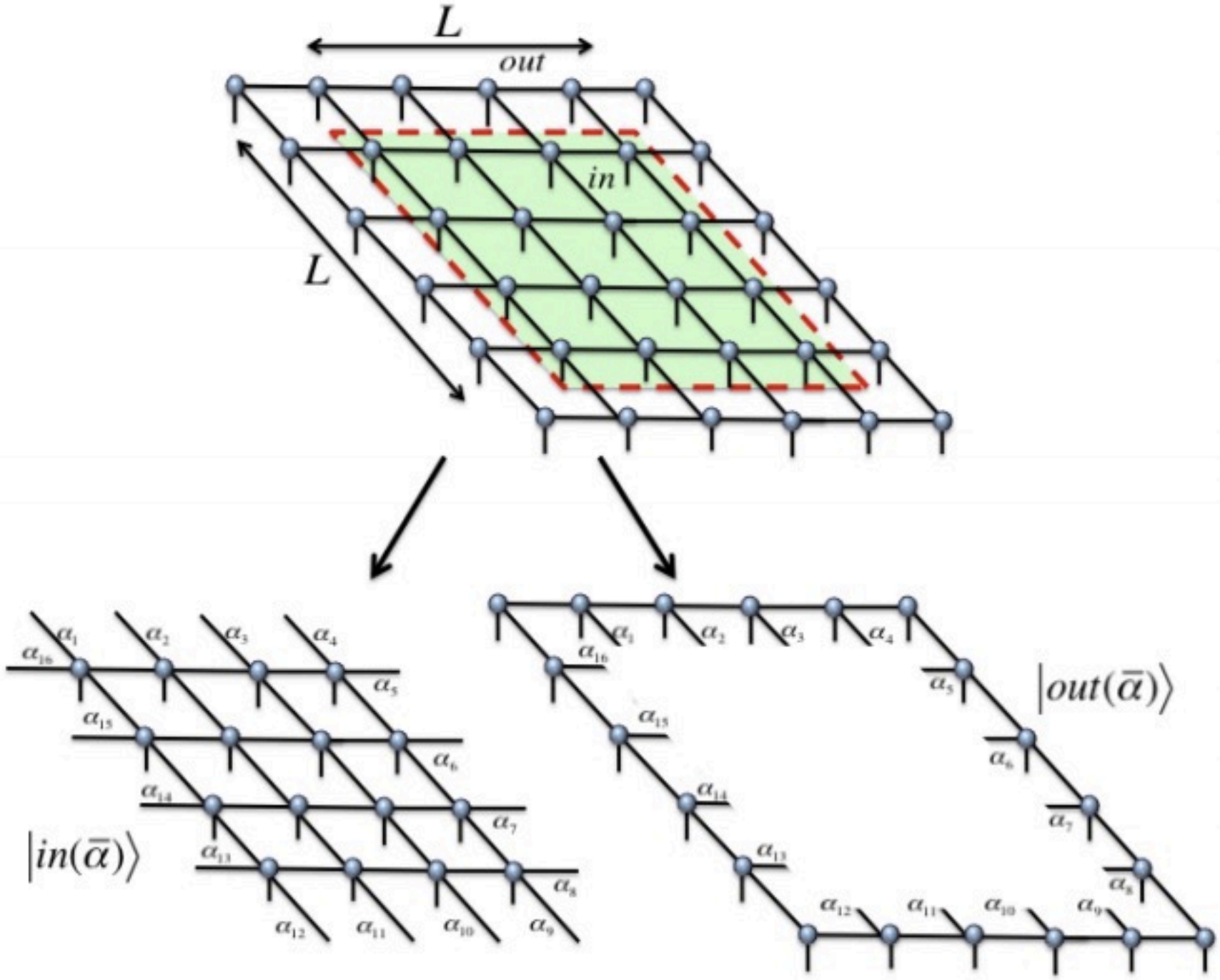
- 二つのテンソル

$$\sum_k A_{ij\ldots k\ldots} B_{i'j'\ldots k\ldots} = C_{ij\ldots i'j'\ldots}$$

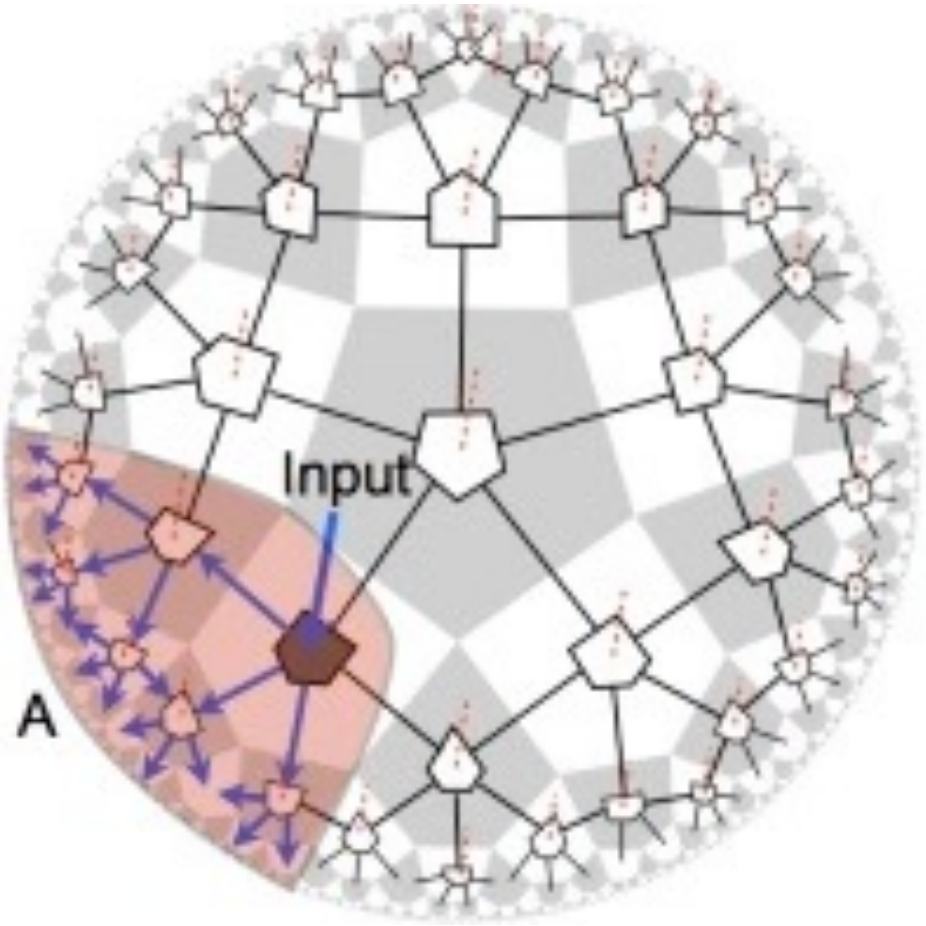


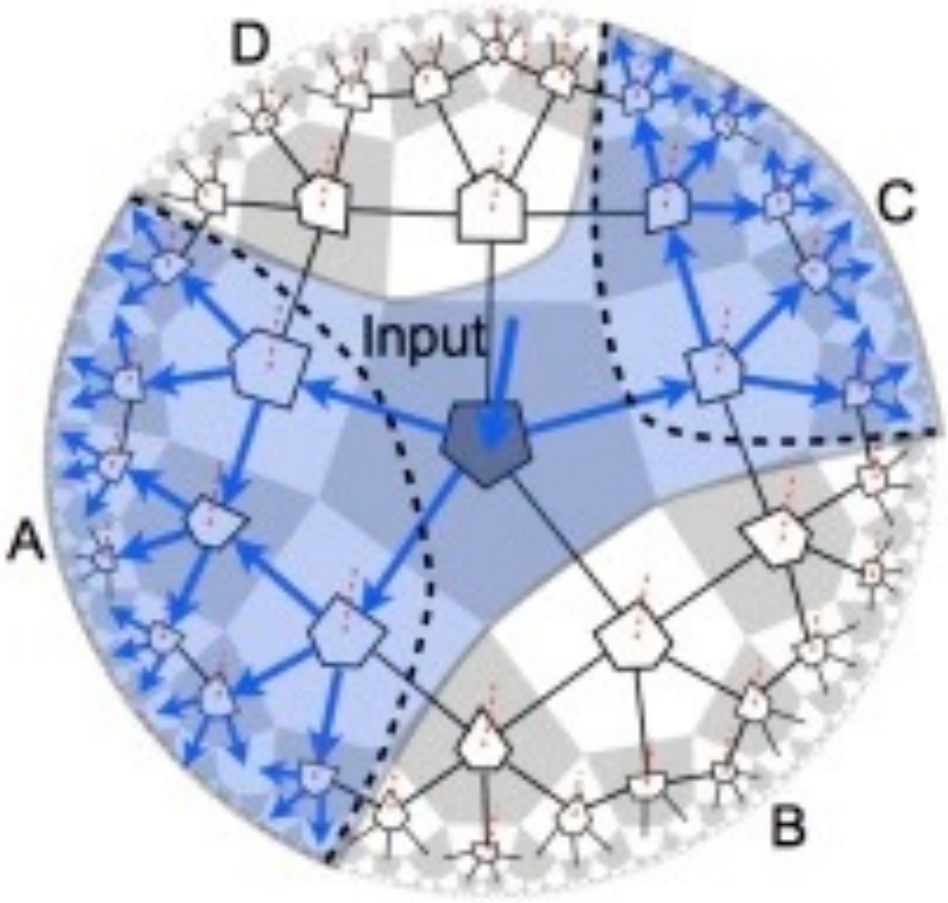
Tensor Networkはテンソルcを、いくつかのパターンに分解する




L
out
in
L
$|in(\bar{\alpha})\rangle$
$|out(\bar{\alpha})\rangle$
α_1 α_2 α_3 α_4 α_5 α_6 α_7 α_8 α_9 α_{10} α_{11} α_{12} α_{13} α_{14} α_{15} α_{16}

AdS/CFT対応とTensor Network(MERA)





https://goo.gl/GUCedR

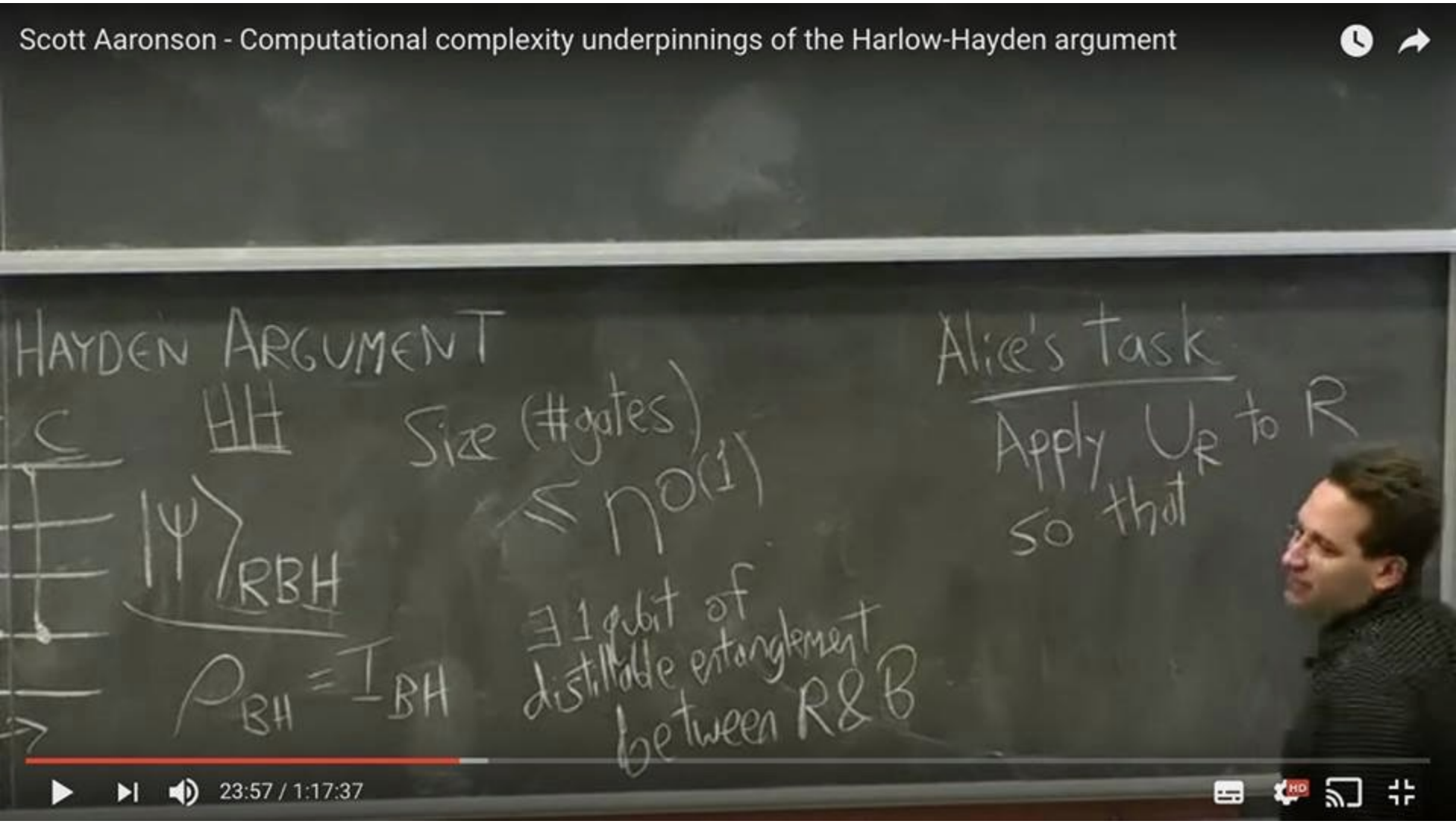
AMPS Paradox 2012

Scott Aaronson

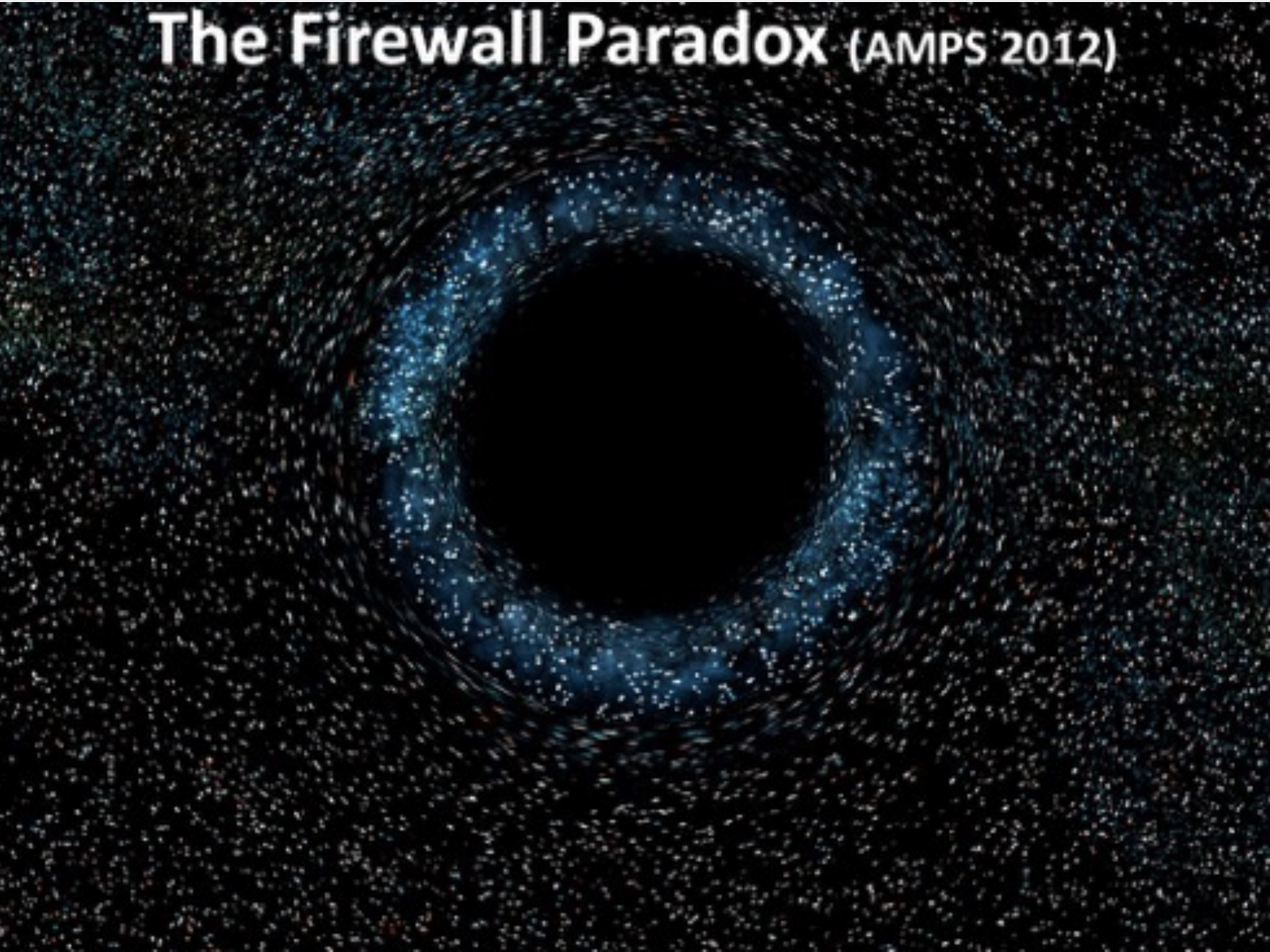
“The Cryptographic Hardness of Decoding Hawking Radiation”

https://www.scottaaronson.com/talks/hawking.ppt

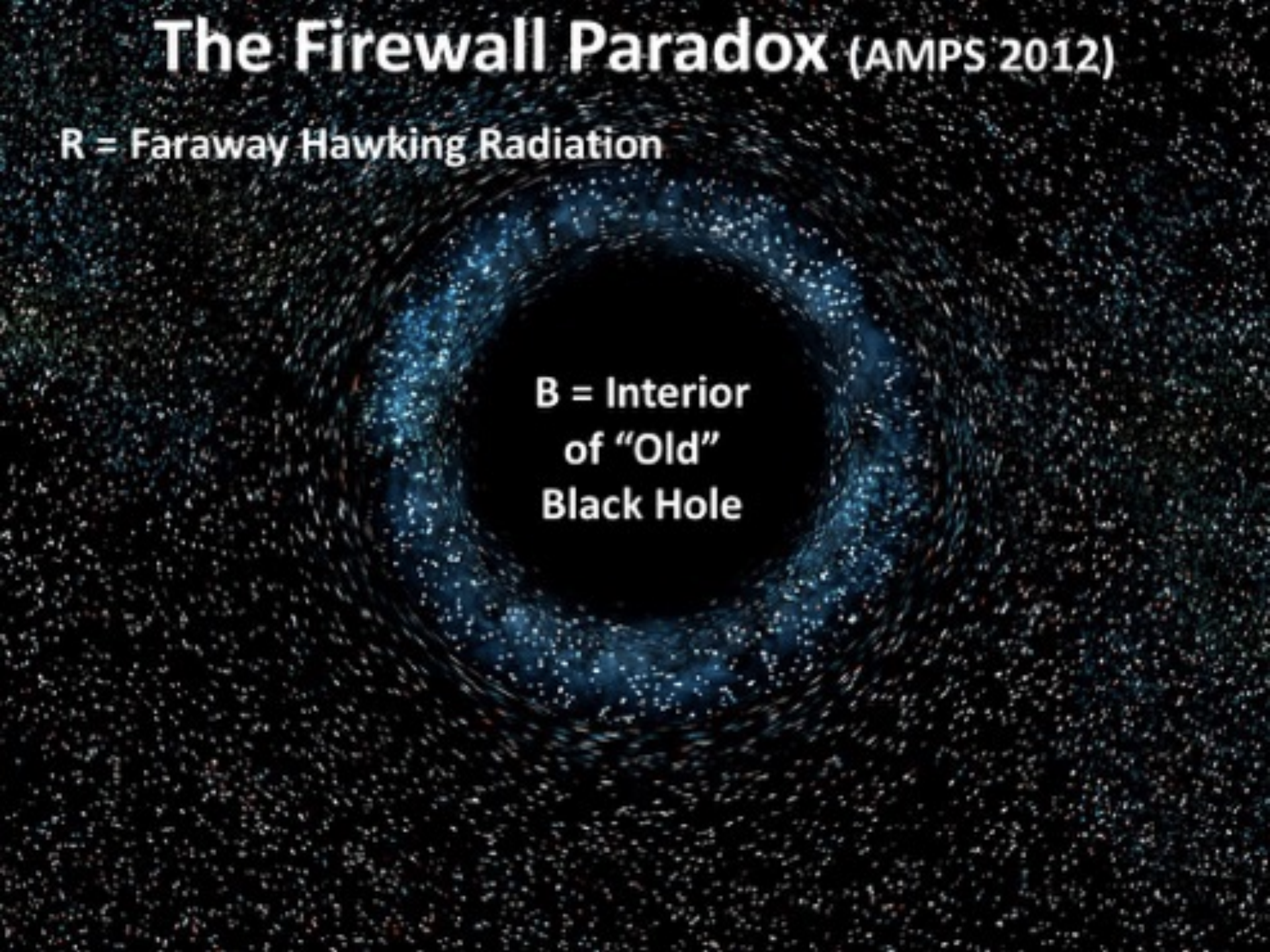
2014年4月


Scott Aaronson - Computational complexity underpinnings of the Harlow-Hayden argument
HAYDEN ARGUMENT
C
HH
$|\psi\rangle_{RBH}$
$\rho_{BH} = I_{BH}$
Size (#gates) $\leq n^{O(1)}$
$\exists$ 1 qubit of distillable entanglement between R&B
Alice's task
Apply U_R to R so that
23:57 / 1:17:37

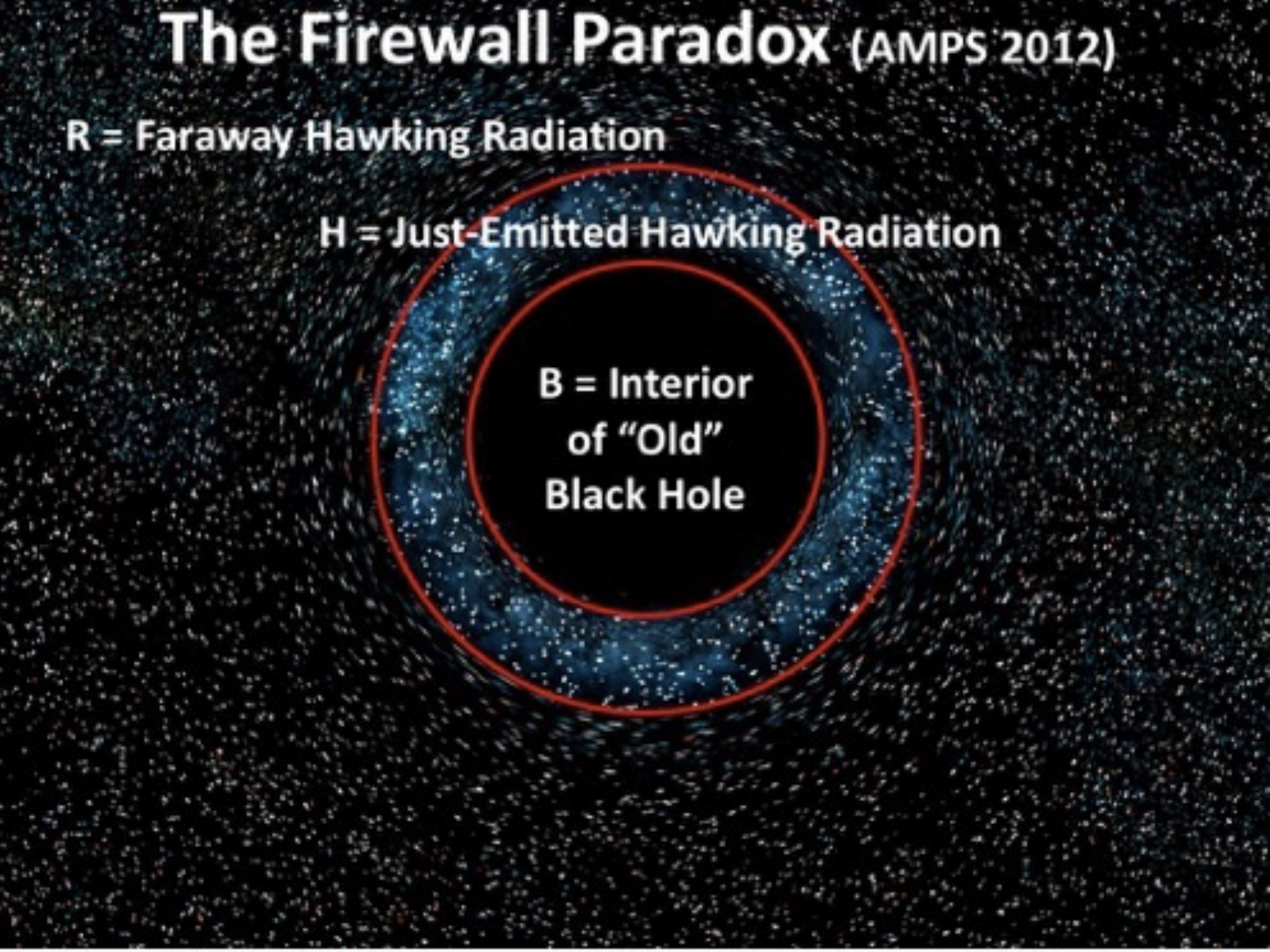
The Firewall Paradox (AMPS 2012)



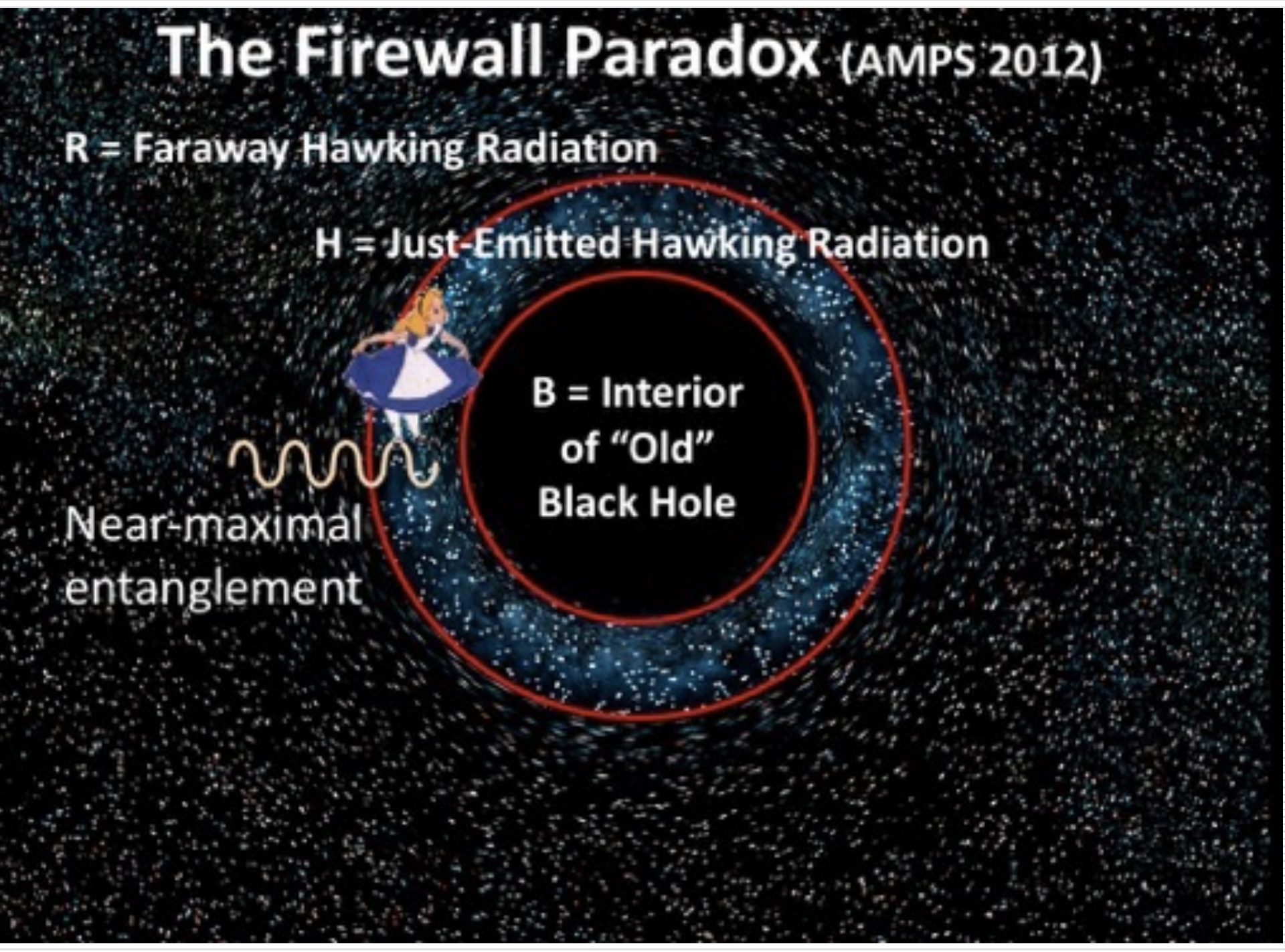
The Firewall Paradox (AMPS 2012)



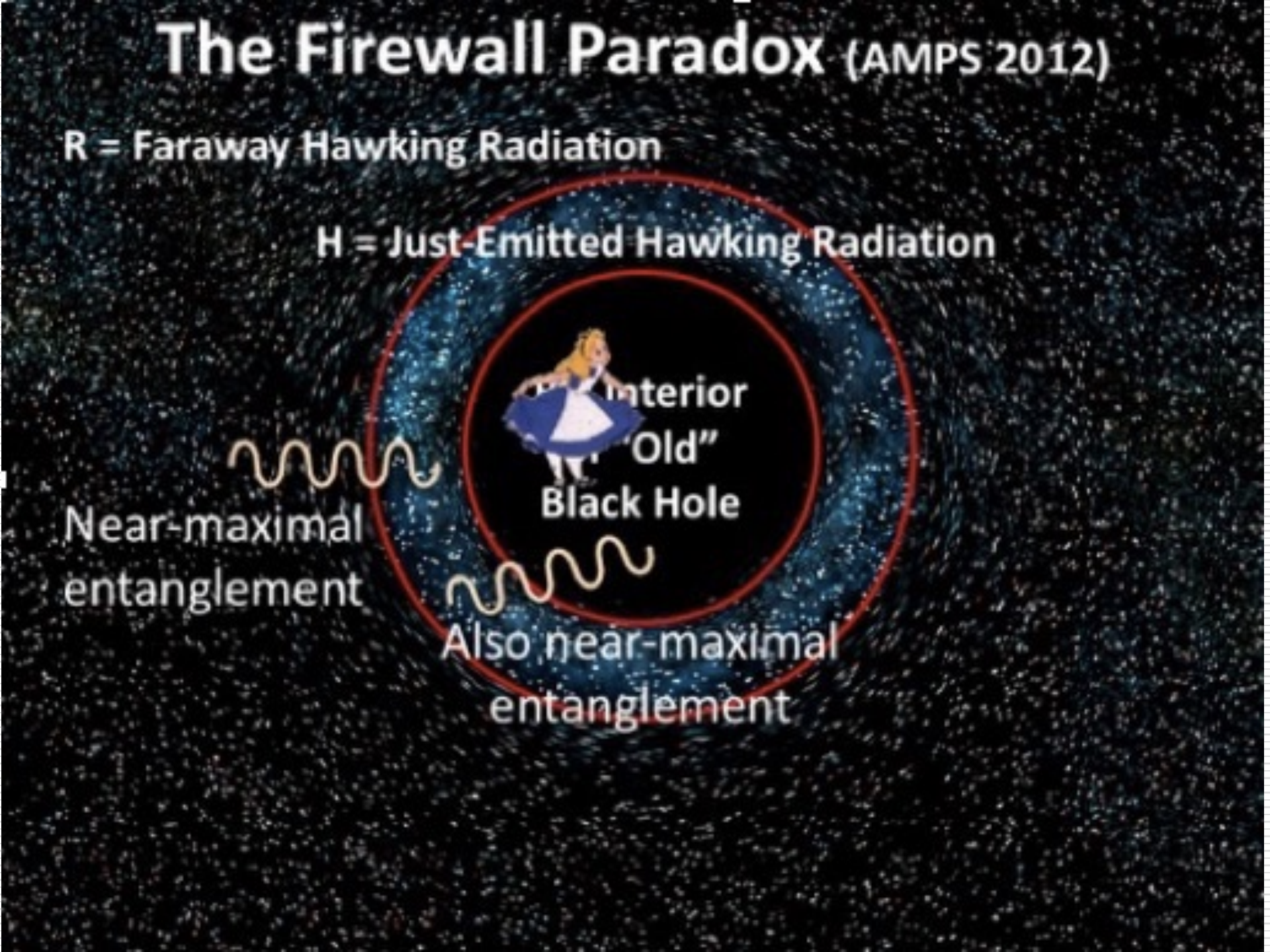
The Firewall Paradox (AMPS 2012)



The Firewall Paradox (AMPS 2012)



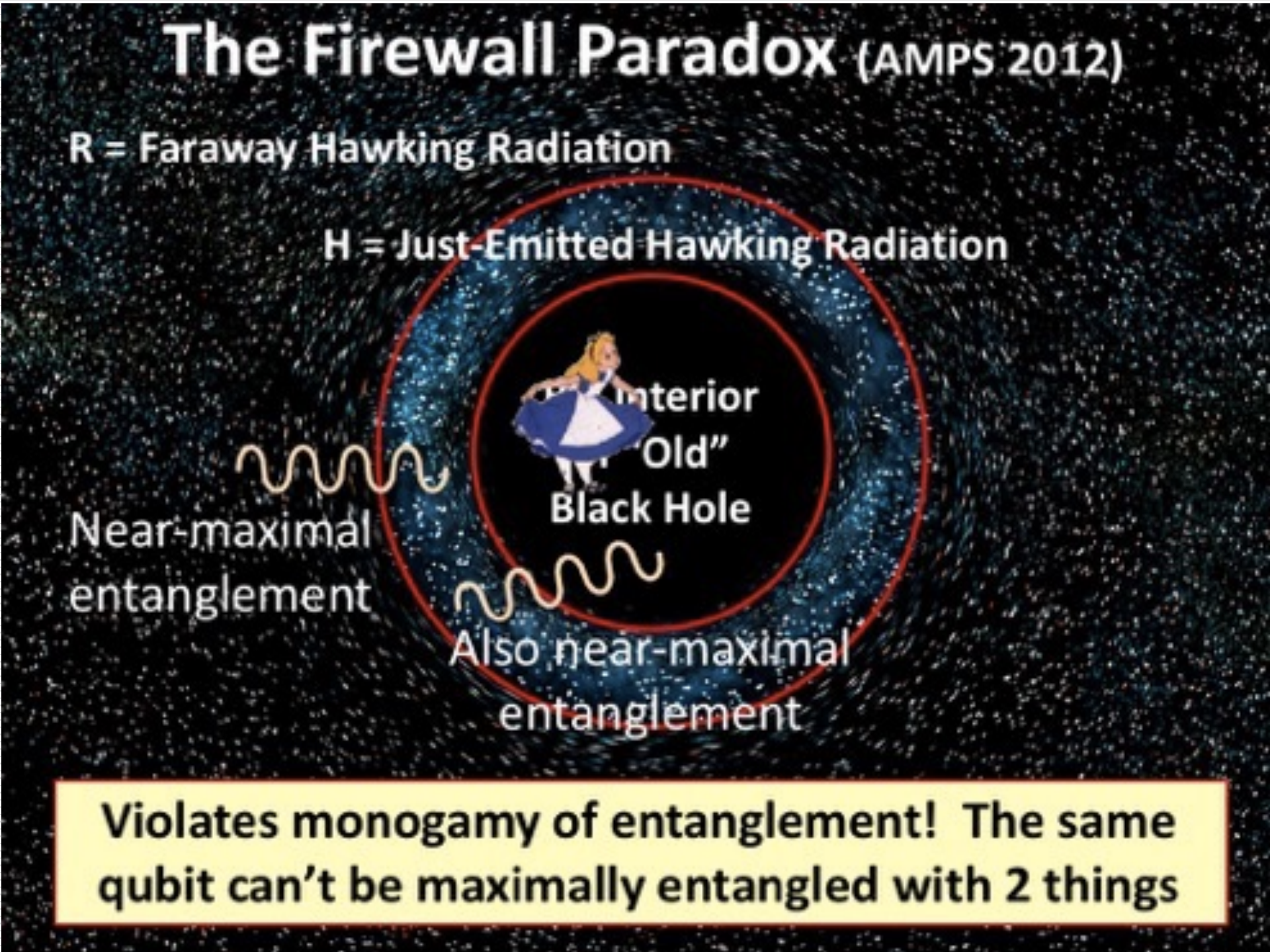
The Firewall Paradox (AMPS 2012)



The Firewall Paradox (AMPS 2012)

R = Faraway Hawking Radiation

H = Just-Emitted Hawking Radiation

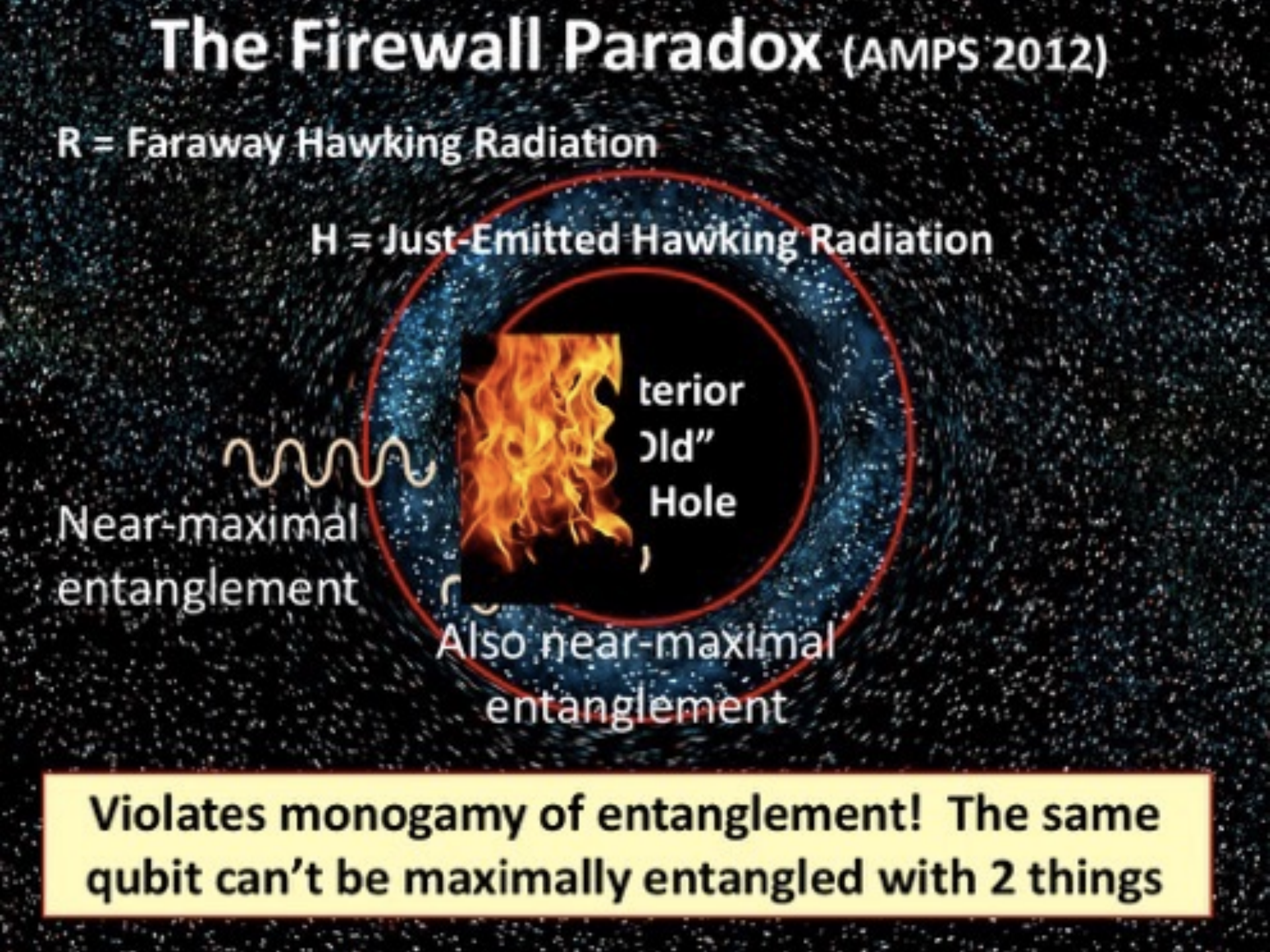


Violates monogamy of entanglement! The same qubit can't be maximally entangled with 2 things

The Firewall Paradox (AMPS 2012)

R = Faraway Hawking Radiation

H = Just-Emitted Hawking Radiation



Violates monogamy of entanglement! The same qubit can't be maximally entangled with 2 things

ER=EPR

Juan Maldacena & Leonard Susskind
"Cool horizons for entangled black holes"
https://arxiv.org/pdf/1306.0533.pdf
2013年7月

1935年のアインシュタインの二つの論文
EPR論文　エンタングルメントの発見

- 「神はサイコロを振らない」
  アインシュタインが、量子論に懐疑的だったことは、よく知られていると思う。量子論の否定は、現在の我々の目から見れば、明らかに、アインシュタインの「誤り」と言っていいものだ。
- 1935年に、アインシュタインとポドルスキーとローゼンは、概略、次の論文を発表する。（三人の著者の頭文字をとって、EPR論文と呼ばれる。）彼が、次の論文で量子論の中に見つけた「矛盾」は、当時の物理学者の誰も気がつかなかった深いものだった。
- "Can Quantum-Mechanical Description of Physical Reality Be Considered Complete ?"「物理的な実在の量子力学の記述は、完全なものと考えることができるか？」 https://goo.gl/qAWacP

1935年のアインシュタインの二つの論文
EPR論文　エンタングルメントの発見

- アインシュタインが発見した、量子力学の「矛盾」を、「EPRパラドックス」という。これに「エンタングルメント」という名前をつけたのは、シュレディンガーらしい。
- 今日では、エンタングルメントは、パラドックスどころか、量子論の基礎を構成するもっとも基本的な前提となっている。
- ただ、このエンタングルメントという奇妙な現象を、誰よりも先に発見したのは、他ならぬアインシュタインであったことは、興味ふかいことである。

EPR論文を報ずる1935年のニューヨークタイムズ紙

EINSTEIN ATTACKS QUANTUM THEORY

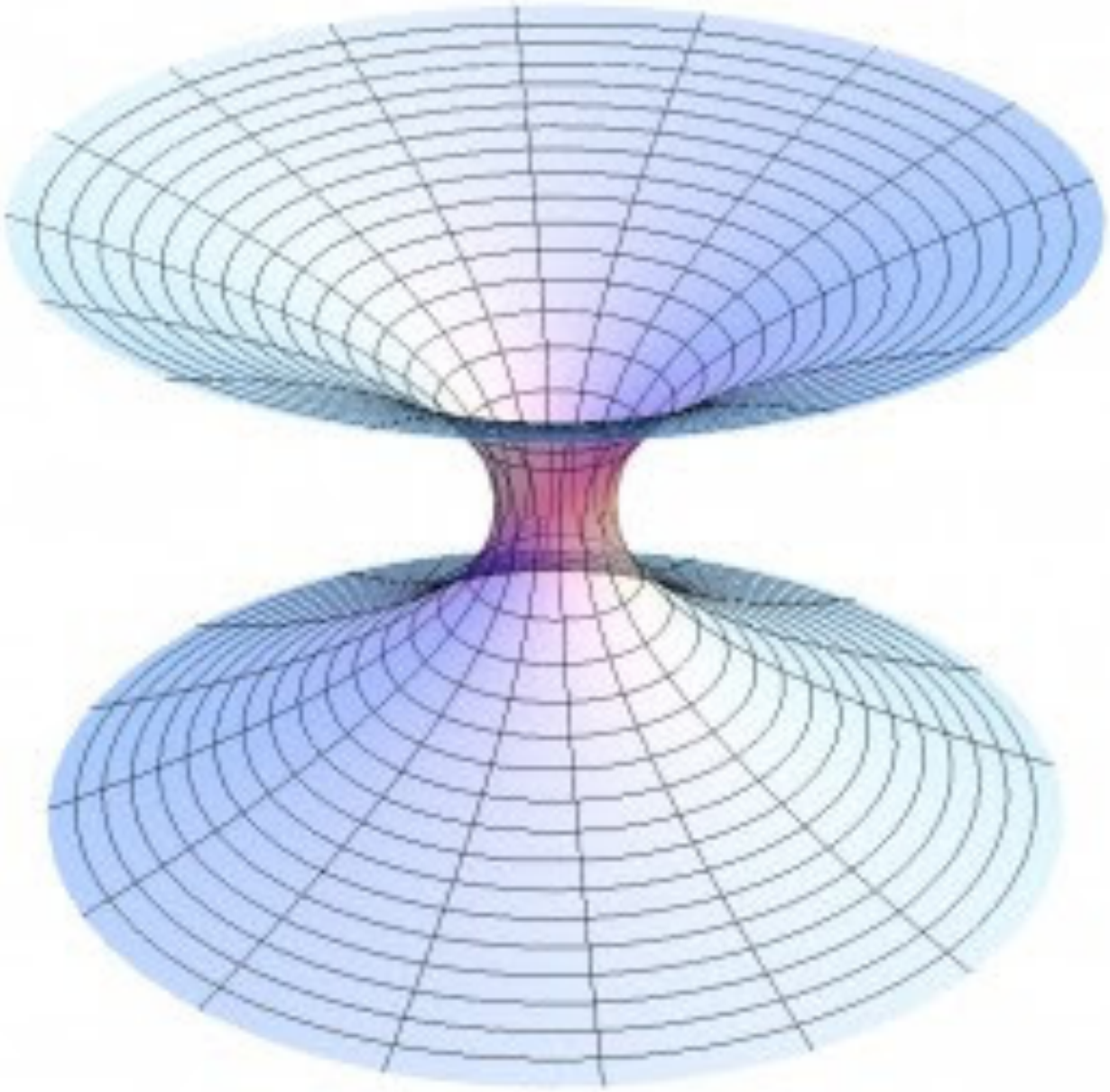
Scientist and Two Colleagues Find It Is Not 'Complete' Even Though 'Correct.'

SEE FULLER ONE POSSIBLE

Believe a Whole Description of 'the Physical Reality' Can Be Provided Eventually.

1935年のアインシュタインの二つの論文
ER論文　ブラックホールのワームホール

- アインシュタインは、先のEPR（Einstein, Podolsky, Rosen）論文を5月に公開してすぐに、ローゼンと一緒に、次の論文を公開する。1935年7月のことだ。この論文をER論文と呼ぶことがある。
- "The Particle Problem in the General Theory of Relativity"「相対性の一般理論における粒子の問題」 https://goo.gl/KG4mjp
- 一般には、二つのブラックホールを結ぶ「橋」が存在しうることを発見した論文だと言われている。この「橋」は、"Einstein-Rosen Bridge" と呼ばれる。別名「ワームホール」とも呼ばれる。（ただ、"wormhole" でWeb をググっても、あまりスジのいい情報は引っかからない。）



1935年のアインシュタインの二つの論文
ER論文とEPR論文には関連があるのか？

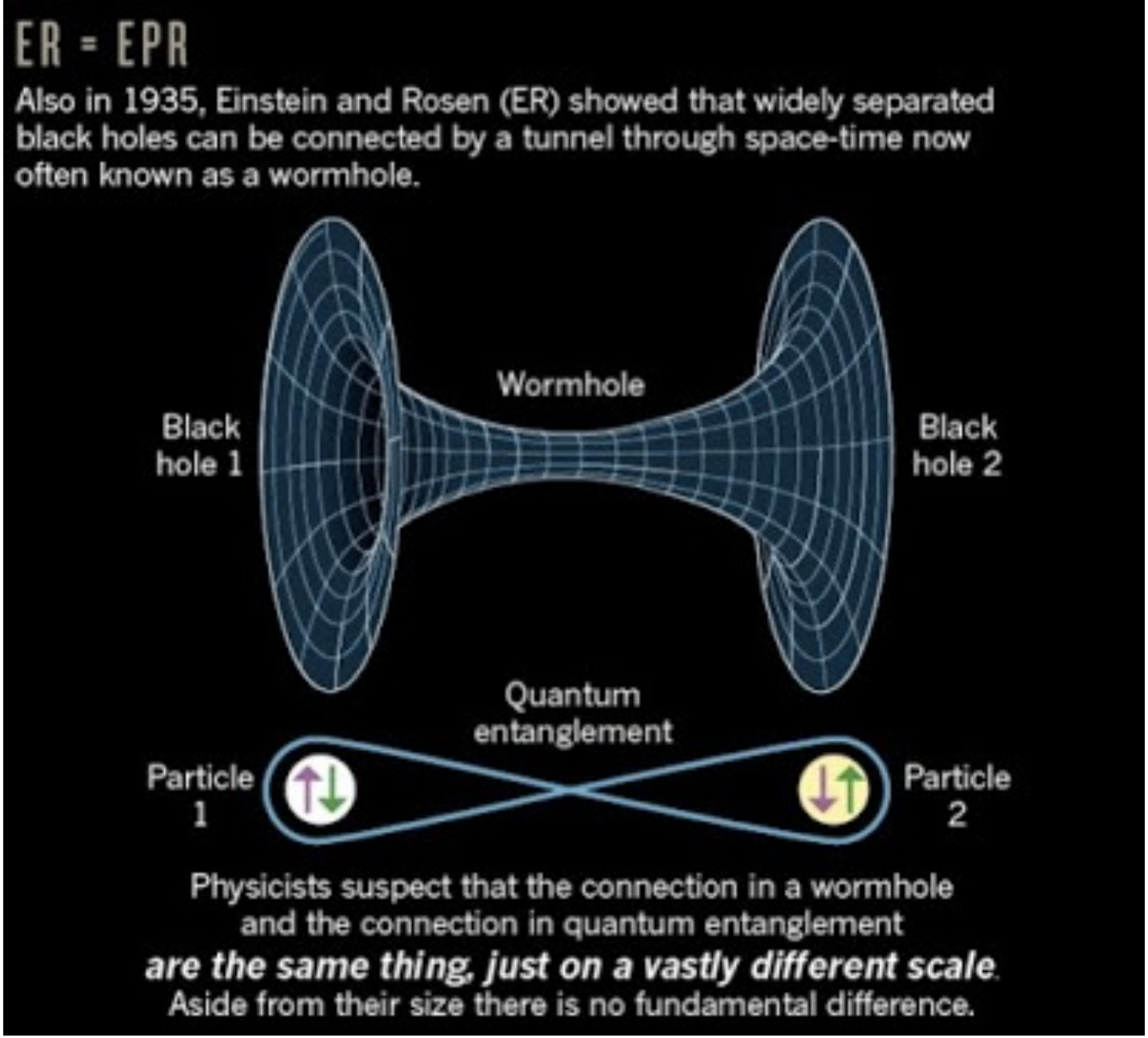
- 1935年に、アインシュタインは、「エンタングルメント」（ただし、パラドックスとして）と「ワームホール」を発見しているのである。ほぼ同時期に行われた、この二つの発見に何か関連があるのだろうか？

- 80年前になされるべきこうした問いかけを、現代に蘇らせたのは、マルダセナ（Maldacena、AdS/CFT 対応の発見者）とサスキンド（Susskind）だった。2013年の論文 "Cool horizons for entangled black holes" https://goo.gl/wU1pKK で、二人は、大胆な仮説を提示する。

ER＝EPR仮説

- 「二つのブラックホールの間のアインシュタイン・ローゼン・ブリッジは、二つのブラックホールのミクロな状態のEPR状の相関関係（エンタングルメント）によって生成される。 … 我々は、これを ER ＝ EPR 相関関係と呼ぶ。別の言葉で言えば、ERブリッジは、その中では、量子システムに関連するEPRが、アインシュタインの重力の記述に弱く結合している特殊なタイプのEPR相関関係なのである。」

- 要するに、1935年にアインシュタインが発見した「エンタングルメント」と二つのブラックホールを結ぶ「アインシュタイン・ローゼン・ブリッジ」は、スケールが全く違うのだが同じものだというのである。これを、「ER ＝ EPR仮説」と呼ぶ。



Natureの2015年11月の記事 "The quantum source of space-time”
https://goo.gl/x0HqA2

Wormholes and Entanglement

John C. Baez & Jamie Vicary
Wormholes and Entanglement
https://arxiv.org/pdf/1401.3416.pdf
2014年2月

- Maldacena and Susskind have proposed a correspondence between wormholes and entanglement, dubbed ER=EPR. We study this in the context of 3d topological quantum field theory, where we show that the formation of a wormhole is the same process as creating a particle-antiparticle pair. A key feature of the ER=EPR proposal is that certain apparently entangled degrees of freedom turn out to be the same. We name this phenomenon ‘fake entanglement’, and show how it arises in our topological quantum field theory model.

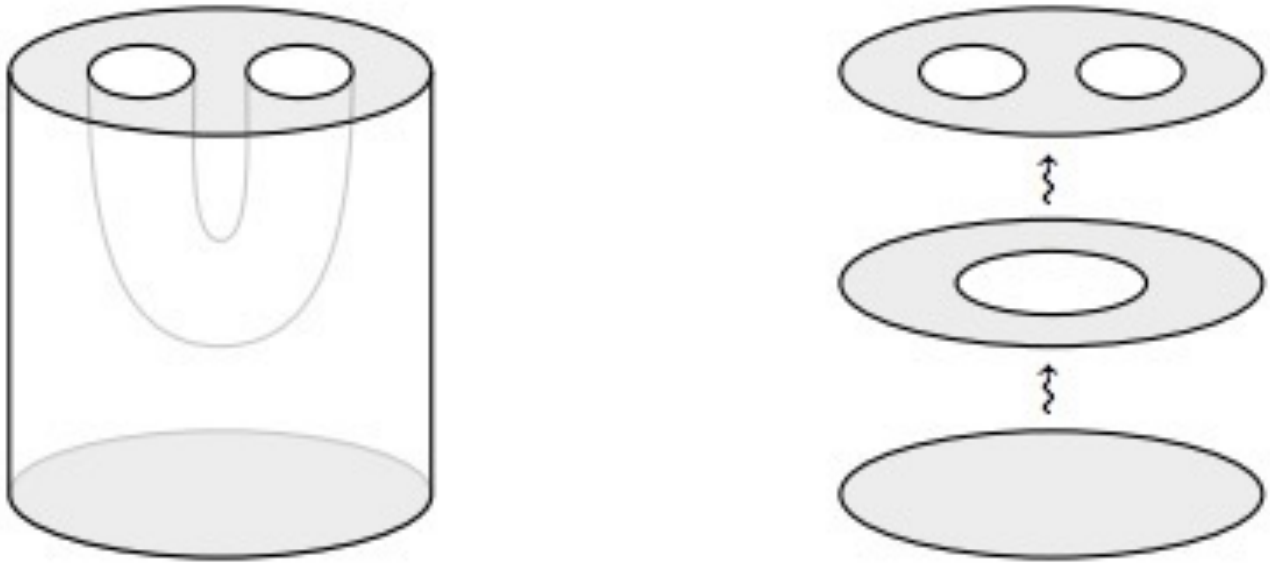


Figure 2: Spacetime history of pair creation in a 3d TQFT

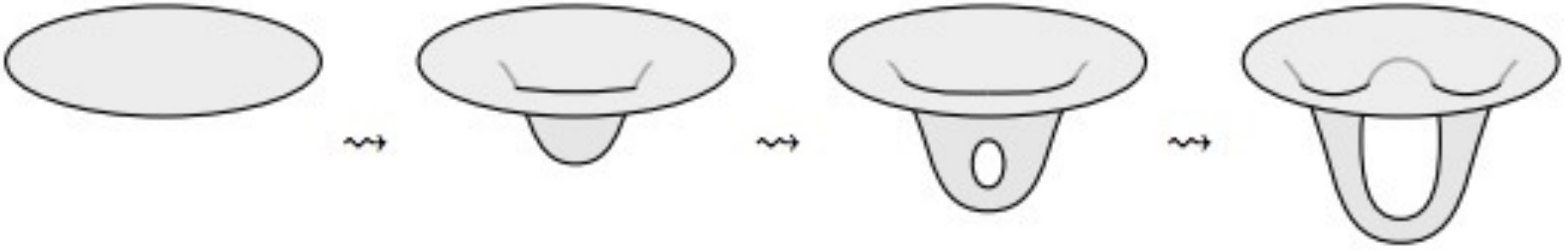
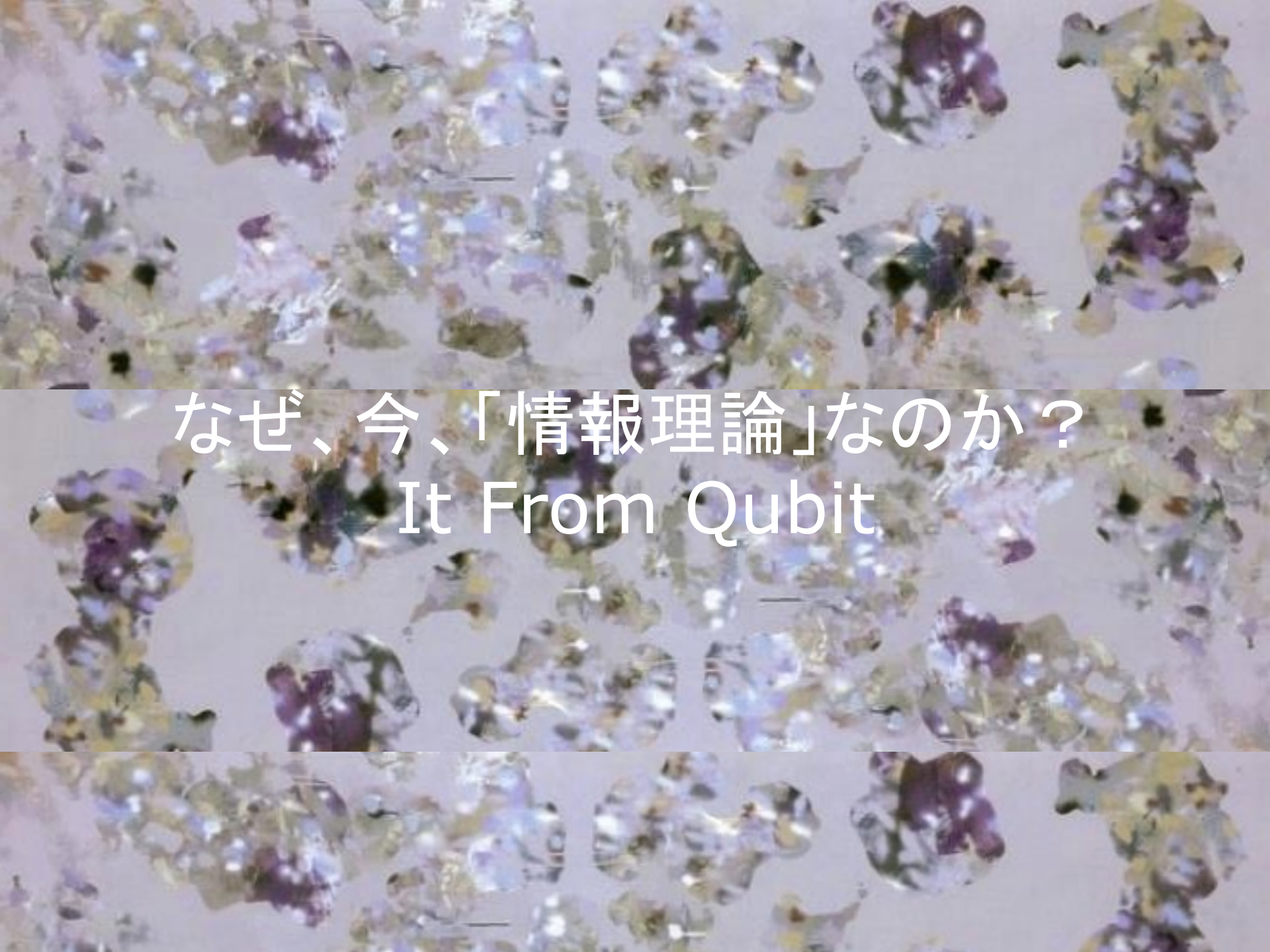


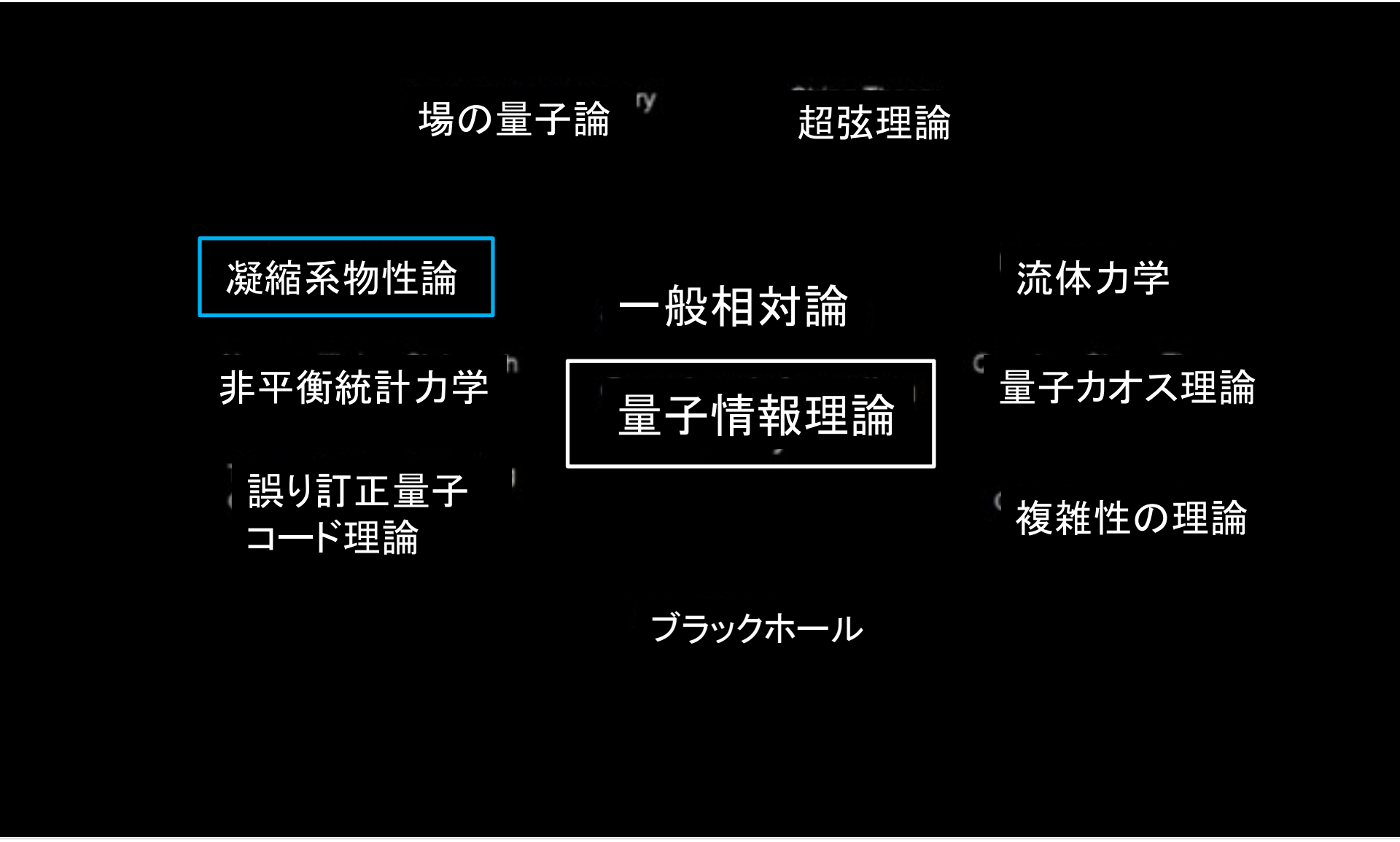
Figure 3: Formation of a wormhole


なぜ、今、「情報理論」なのか？
It From Qubit

物理学者は、
現代の物理学をどのように捉えているのか？

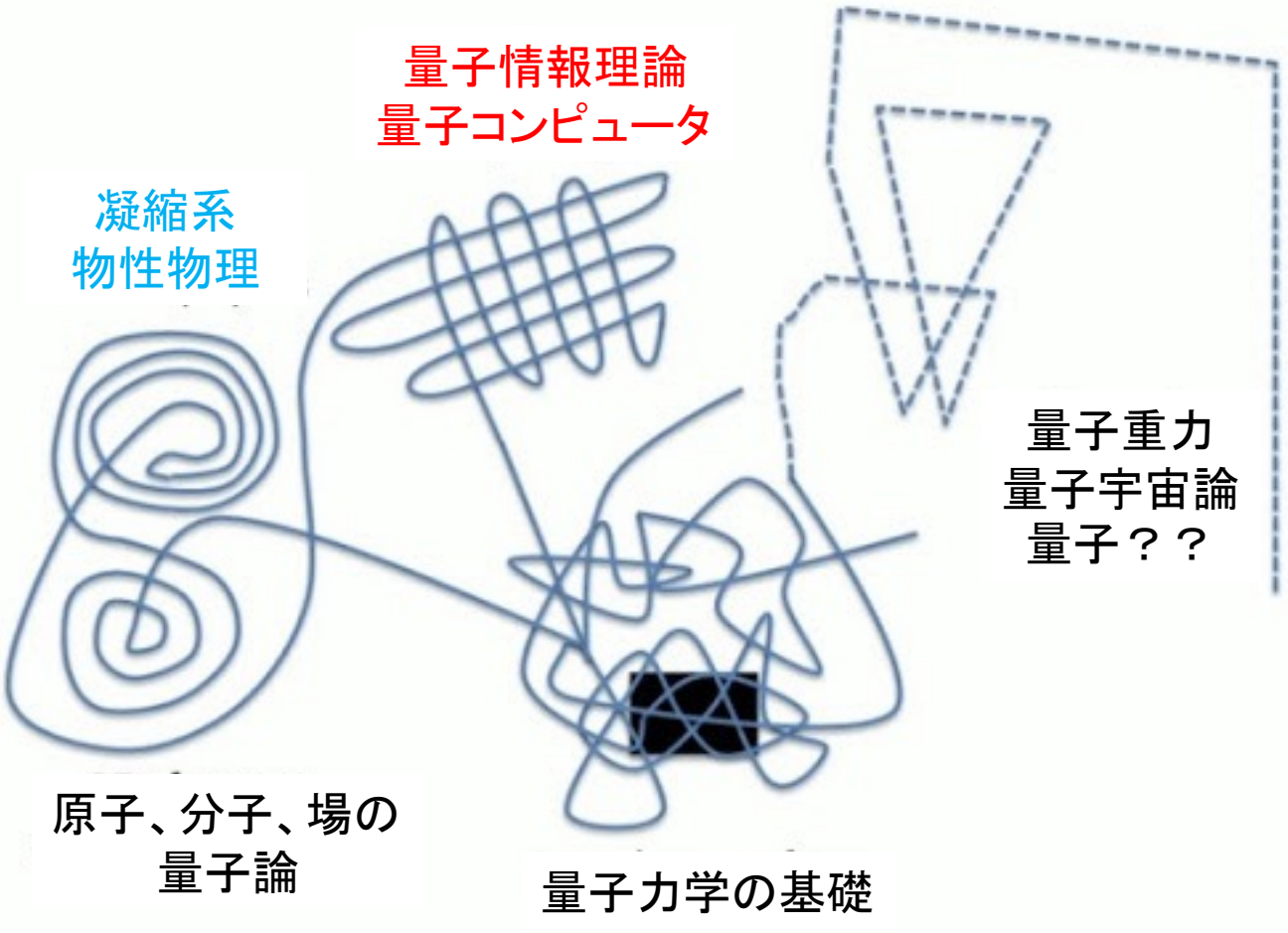
ここでは、現代の物理学者が、現代物理学の課題を、どのように捉えているのかを見ておこう。

現代物理学の俯瞰図 by Susskind

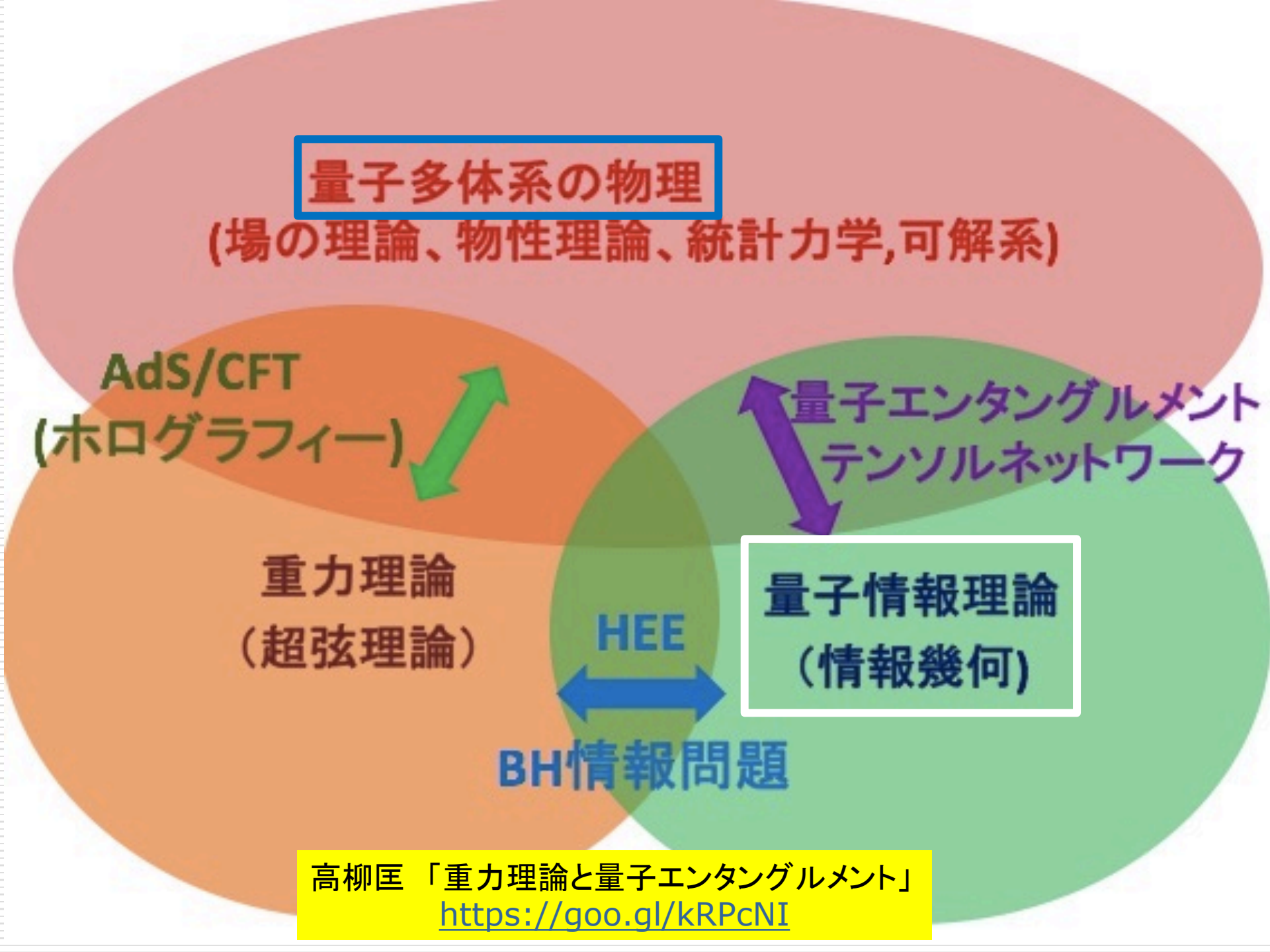


Susskind "Entanglement and Complexity: Gravity and Quantum Mechanics"
https://goo.gl/J0wSkf

エンタングルメントの歴史の絵解き

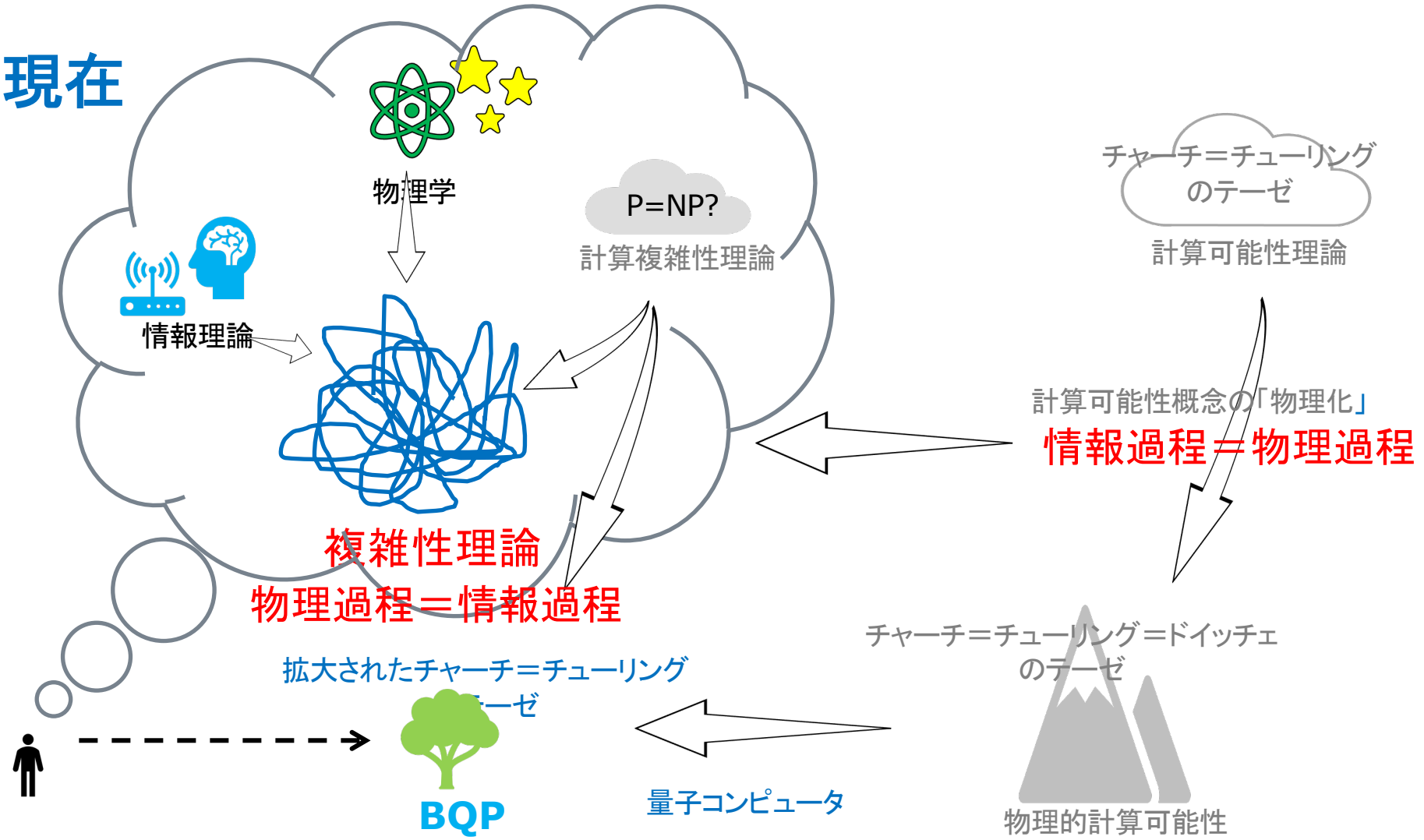


"Our Quantum Society: Living with Entanglement" https://goo.gl/aWtAzi
Cathryn Carson https://goo.gl/j7bE57



高柳匡 「重力理論と量子エンタングルメント」
https://goo.gl/kRPcNI

現在



計算の「限界」は、我々の身近の具体的な問題に近づいてきている。物理学の複雑性理論の発展もめざましい。

Part V

数学の「証明」をめぐって

Agenda Part V

数学の「証明」をめぐって

- 数学の「証明」の問題
- 数学でのコンピューターの利用
- Voevodsky
- Univalent FoundationsとUniMath
- 数学とNP問題
- 複雑性理論でのクラス MA: Merlin-Arthur

数学の「証明」の問題

誤っていた「証明」

- 「フェルマーの定理」は、最終的には、1995年のAndrew Wilesの論文によって証明されたが、フェルマー自身は、この定理を証明出来たと信じていた。それが、誤った「証明」であったのは確かである。
- Wiles自身も、95年の論文以前に一度、「証明した」とする発表を行うが、誤りが見つかり、それを撤回している。
- 世紀の難問である「リーマン予想」は、何度か「証明」が発表されている。今までのところ、それは誤って「証明」だった。

正しいことの判定が難しい証明

- Grigory Perelmanは、2002年から2003年にかけて arXiv に投稿した論文で、「三次元のポアンカレ予想」を解いたと主張した。
- arXivは、学会の論文誌とは異なり、掲載の可否を決める査読が基本的にないので、発表の時点では、彼の論文以外に、彼の証明が正しいという裏付けはなかった。
- 彼の証明の検証は、複数の数学者のグループで取り組まれ、最終的に彼の証明の正しさが「検証」されたのは、2006年のことだった。

非常に膨大な証明

- 「有限単純群の分類問題」は、Daniel Gorensteinをリーダーとする数学者の集団によって解決され、20世紀の数学の大きな成果の一つと呼ばれている。
- ただ、その「証明」は、100人以上の数学者が、50年のあいだに数百の論文誌に発表した、膨大な量の「証明」の総和である。そのページ数は一万ページを超えると言われている。（「全部を読んでる数学者は、1人もいない。」というのがジョークなのか本当なのか、分からない。）

数学でのコンピューターの利用

なぜ、数学の証明にコンピュータが必要なのか？

- フェルマーの定理を証明したワイルズが、謙遜して自分は「巨人の肩の上の小人だ」と言ったように、数学は、先行した無数の人たちの業績の蓄積の上に成り立っている。
- ピタゴラスの「教団」は、いくつかの発見を「秘教」にしていたらしいのだが、もしも、ピタゴラスの子孫が生きていたとしても、ピタゴラスの定理を使うのに、彼らに著作権料を払う必要はない。また、その「証明」を自分で繰り返す必要はない。
- 少なくとも現代の数学では、情報の共有は、学問自体の前提でさえある。

「四色問題」のコンピュータによる解決

- 1974年、イリノイ大学のKenneth AppelとWolfgang Hakenは、当時の最先端のコンピュータを使って、「四色問題」を解いた。
- コンピュータの稼働時間は、1200時間に及び、夜間の空き時間しか利用が認められなかったので半年がかりの計算だったと言う。
- コンピュータが人手にはあまる膨大な計算を行ったのは事実だが、当時、その計算の正しさの検証しようのないことを理由に、こうした「証明」を疑問視する声も一部にはあった。

「四色問題」「Feit-Thompsonの定理」のCoqによる証明

- 2004年、Georges Gonthierは、Coqを使って「四色問題」を解いた。
- Georges Gonthierは、また、2013年、有限群分類の重要な定理である、「Feit-Thompsonの定理」のCoqによる形式的証明を与えた。
- 15,000の定義、4,300の定理、17万行のソースからなるこの証明を、彼はチーム作業で、6年かけて完成させた。

計算結果を印刷すると、
マンハッタン島ぐらいの
サイズになる

Mathematicians Map E_8

Mathematicians have mapped the inner workings of one of the most complicated structures ever studied: the object known as *the exceptional Lie group E_8*. This achievement is significant both as an advance in basic knowledge and because of the many connections between E_8 and other areas, including string theory and geometry. The magnitude of the calculation is staggering: the answer, if written out in tiny print, would cover an area the size of Manhattan. Mathematicians are known for their solitary work style, but the assault on E_8 is part of a large project bringing together 18 mathematicians from the U.S. and Europe for an intensive four-year collaboration.

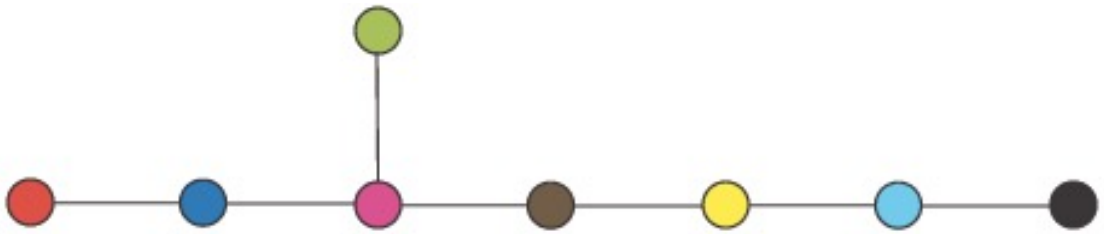
http://www.aimath.org/E8/

人間の遺伝子情報は、1ギガ以下。
E8の計算結果は60ギガ。

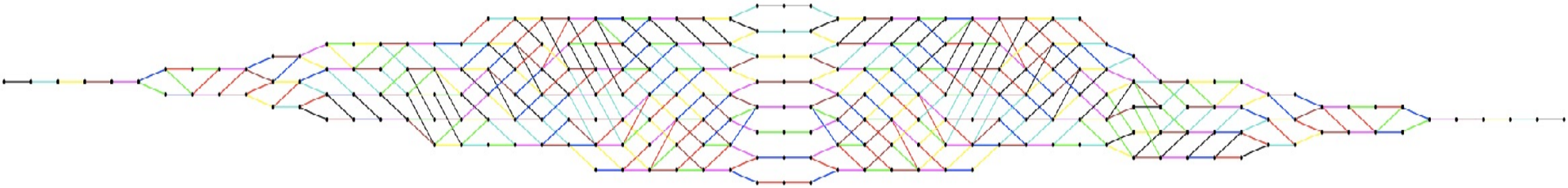
E_8 Map: Bigger than the Human Genome

The magnitude of the E_8 calculation invites comparison with the Human Genome Project. The human genome, which contains all the genetic information of a cell, is less than a gigabyte in size. The result of the E_8 calculation, which contains all the information about E_8 and its representations, is 60 gigabytes in size. That is enough space to store 45 days of continuous music in MP3 format. While many scientific projects involve processing large amounts of data, the E_8 calculation is very different: the size of the input is comparatively small, but the answer itself is enormous, and very dense.

E_8 Dynkin diagram

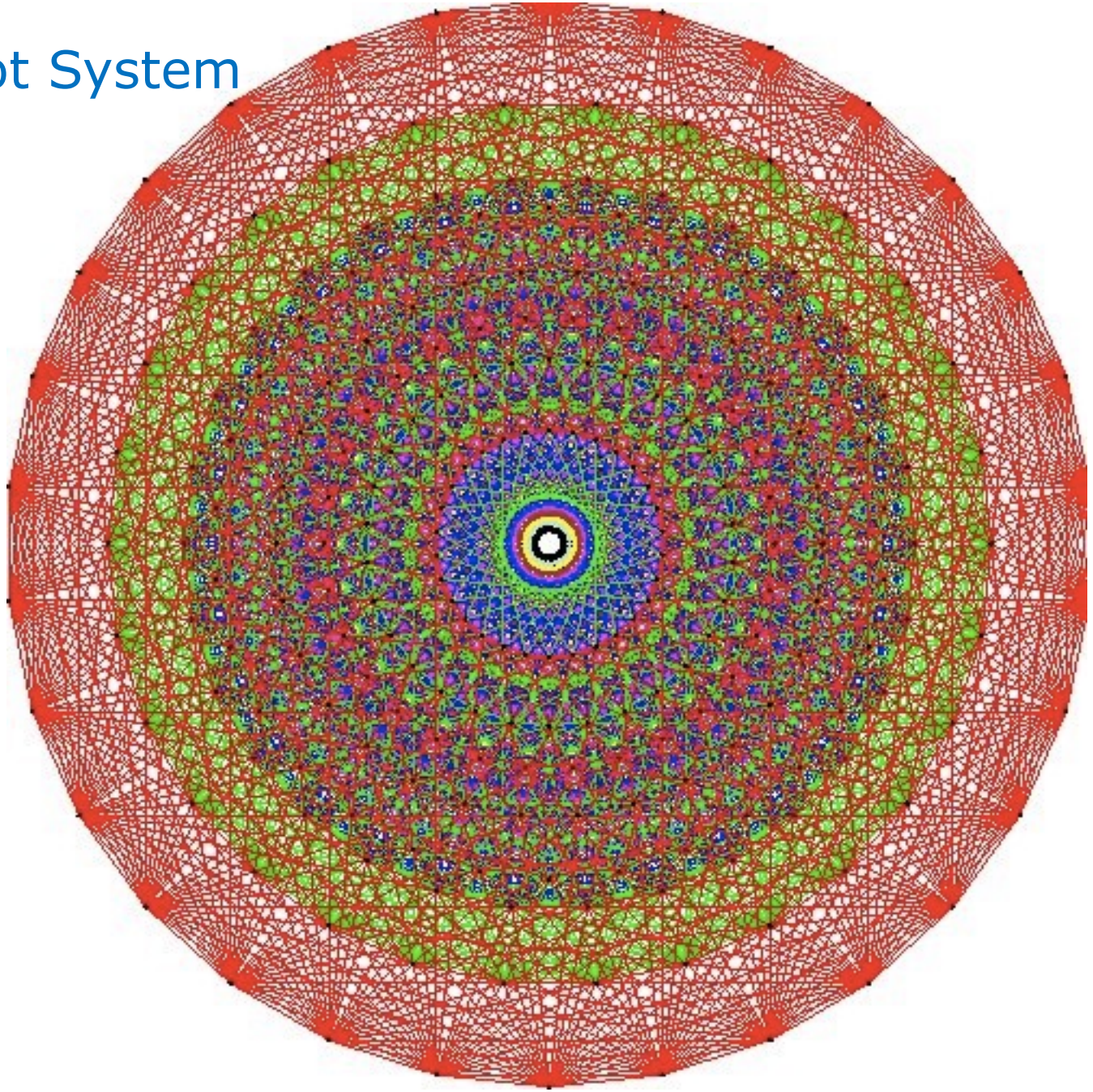


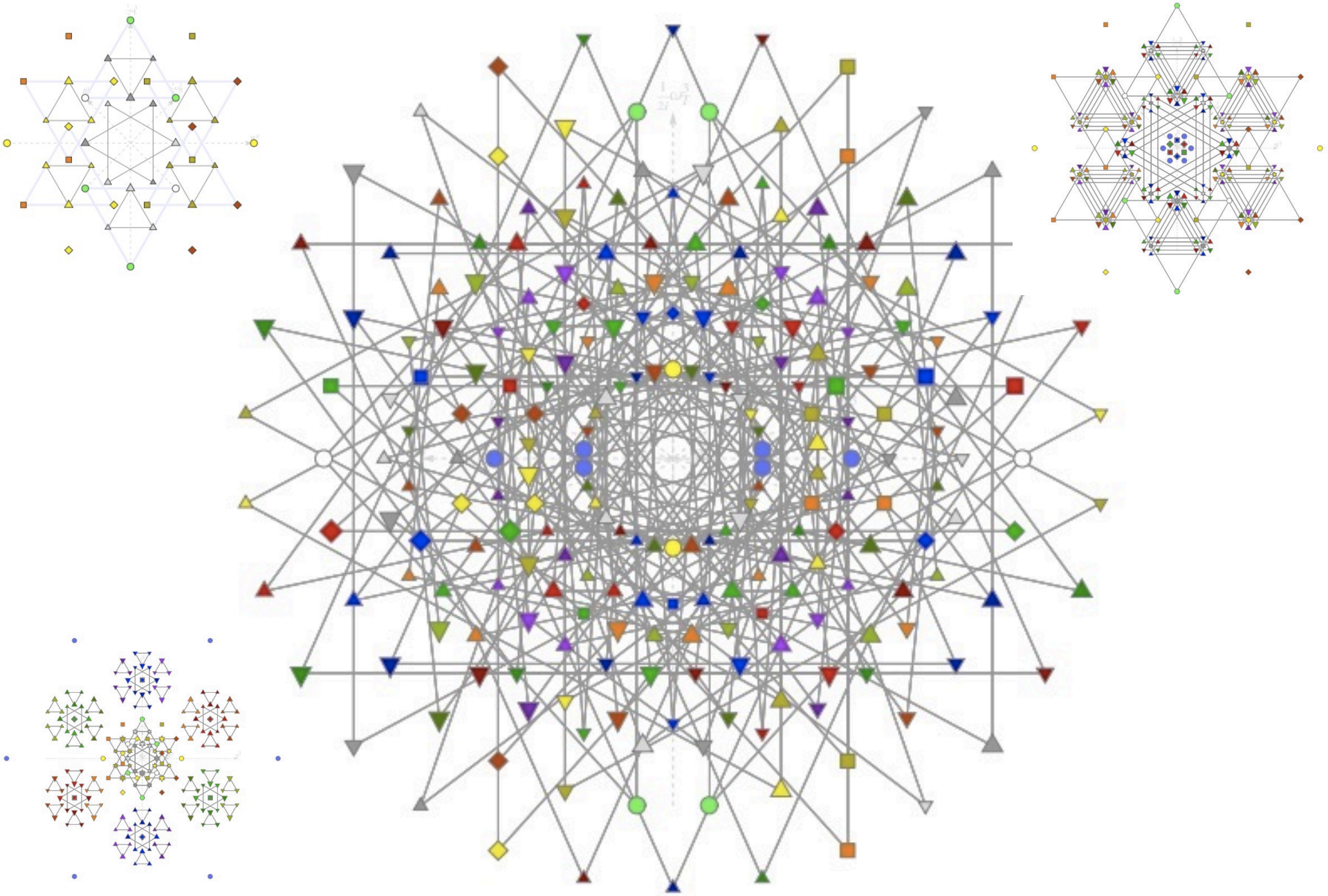
picture of the 248-dimensional Lie algebra of E_8



The Lie algebra E_8 is generated by 16 elements X_{red}, Y_{red}, X_{blue}, Y_{blue},..., X_{black}, Y_{black} corresponding to the nodes in the Dynkin diagram.

E_8 Root System





Garrett Lisi: An Exceptionally Simple Theory of Everything
https://goo.gl/bmfwxi

Voevodsky

Voevodsky
1966年6月4日 - 2017年9月30日



Voevodsky

- 昨年亡くなったVoevodskyは、Milner予想、Bloch-Kato予想を解くなど、代数幾何でグロタンディックが進もうとした道で、大きな業績を残した。ヴョブドスキーの最後の仕事は、数学の基礎とコンピュータに関係していた。
- 彼は、数学の証明に、コンピュータを使うべきだと主張した最初の数学者の一人で、また、そのためのコンピュータによる証明支援システムのライブラリーUniMthを開発した。
- GitHub: https://github.com/UniMath/UniMath
- 2016年9月の講演 "UniMath - a library of mathematics formalized in the univalent style" https://goo.gl/3sJr1M

Voevodskyの経験

- 2000年頃、彼は1993年に自分が発表した論文の重要な補題が間違っていたことに気づく。でも、その頃には、その論文は広く出回っていて、多くの数学者がその証明を「信じて」いた。彼が、その間違った補題なしでも、論文の結論が正しいことを証明できたのは、2006年になってからだった。
- 別のこともあった。1998年に共著で彼が発表した論文の証明に対して、「正しくない」という批判が出される。結論的には、彼は、正しかったのだが、彼が、最終的に、自分が正しいことを確信できたのは、2013年になってからだった。
- （このあたりの経緯は、"The Origins and Motivations of Univalent Foundations" https://goo.gl/LW2Wcq に、詳しく触れられている。）

- ヴョブドスキーは、考える。「数学が、累積的（accumulative）な性格を持つのなら、もしも、そこに誤りが紛れ込むと、それも、累積する可能性がある。」
- ワイルズのフェルマーの定理の1993年の証明には、誤りがあった。それが修正されたのは、1995年のことだ。どんどん複雑化して高度化する数学の「証明」の正しさをチェックするのは難しい。数学者の「証明」が正しいという保証はないのだ！
- ヴョブドスキーは、数学の証明は、コンピュータでチェックできるプログラムの形を取るべきだと主張し、実際に、それを実行してみせた。
- この流れは、21世紀の数学の形式を、大きく変えていくだろう。

Univalent FoundationsとUniMath

https://github.com/UniMath/UniMath

UniMath by Voevodsky https://goo.gl/ShdXzr

Univalent TheoryとCoq

- Voevodskyは、HoTTを武器に、Univalent Theoryという数学の新しい基礎付け・再構成を始めている。興味深いのは、彼が、こうした理論展開を、数学論文ではなく、Coqのプログラムの形でGitHubで公開していることである。
  https://github.com/vladimirias/Foundations/
- 次の論文は、型の理論やHoTTにあまりなじみのない一般の数学者向けの、Voevodskyの理論Coqライブラリーの解説である。
  http://arxiv.org/pdf/1210.5658v1.pdf

UniMath

Today we face a problem that involves two difficult to satisfy conditions.

On the one hand we have to find a way for computer assisted verification of mathematical proofs.

This is necessary, first of all, because we have to stop the dissolution of the concept of proof in mathematics.

On the other hand we have to preserve the intimate connection between mathematics and the world of human intuition.

This connection is what moves mathematics forward and what we oftenexperience as the beauty of mathematics.

https://goo.gl/ShdXzr

The Univalent Foundations (UF) is, a yet imperfect, solution to this problem.

In their original form, the UF combined three components:

- the view of mathematics as the study of structures on sets and their higher analogs,
- the idea that the higher analogs of sets are reflected in the set-based mathematics as homotopy types,
- the idea that one can formalize our intuition about structures on these higheranalogs using the Martin-Lof Type Theory (MLTT) extended with the Law ofExcluded Middle for propositions (LEM) , the Axiom of Choice for sets (AC),the Univalence Axiom (UA) and the Resizing Rules (RR).

The main new concepts that were since added to these are the following:

- the understanding that a lot of mathematics can be formalized in the MLTT without the LEM and the AC and that excluding these two axioms oneobtains foundations for a new form of constructive mathematics,
- the understanding that classical mathematics appears as a subset of this new constructive mathematics,
- the understanding that the MLTT extended with the UA is an imperfect formalization system for this constructive mathematics and that it should bepossible to integrate the UA into the MLTT obtaining a new type theorywith better computational properties.

Library NaturalNumbers

Natural numbers and their properties. Vladimir Voevodsky. Apr. - Sep. 2011

This file contains the formulations and proofs of general properties of natural numbers from the univalent perspecive.

Contents

- Equality and inequality on nat
  - Basic properties of paths on nat and the proofs of isdeceq and isaset for nat
  - S : nat -> nat is a decidable inclusion
- Inequalities on nat
  - Boolean "less of equal" and "greater or equal" on nat
  - Semi-boolean "greater" on nat or natgth
  - Semi-boolean "less" on nat or natlth
  - Semi-boolean "less or equal" on nat or natleh
  - Semi-boolean "greater or equal" on nat or natgeh
  - Simple implications between comparisons
  - Comparison alternatives
  - Mixed transitivities
  - Two comparisons and S
  - Comparison alternatives and S

- Some properties of plus on nat
  - The structure of the additive abelian monoid on nat
  - Addition and comparisons
  - Comparisons and n -> n + 1
  - Two comparisons and n -> n + 1
  - Comparison alternatives and n -> n + 1
  - Cancellation properties of plus on nat
  - Some properties of minus on nat
  - Basic algebraic properties of mul on nat
  - nat as a commutative rig
  - Cancellation properties of mul on nat
  - Multiplication and comparisons
  - Properties of comparisons in the terminology of algebra1.v
  - Submonoid of non-zero elements in nat
  - Division with a remainder on nat
  - Exponentiation natpower n m ("n to the power of m") on nat
  - Factorial on nat
- The order-preserving functions di i : nat -> nat whose image is the complement of one element i.
- The order-preserving functions si i : nat -> nat that take the value i twice.
- Inductive types le with values in UU
  - A generalization of le and its properties
  - Inductive types le with values in UU are in hProp
  - Comparison between le with values in UU and natleh

Equality and inequality on nat

```
Definition natnegpaths (x y : nat) : hProp := hProppair (x != y) (isapropneg _).

Fixpoint natneq_hProp (n m : nat) : hProp :=
  match n, m with
  | S n, S m => natneq_hProp n m
  | O, O => hfalse
  | _, _ => htrue
  end.

Notation " x ≠ y " := (natneq_hProp x y) (at level 70, no associativity) : nat_scope.

Local Open Scope nat_scope.
Lemma negpaths0sx (x : nat) : ¬ (0 = S x).
Proof.
  intro.
  set (f := λ n : nat, match n with O => true | S m => false end).
  apply (negf (@maponpaths _ _ f 0 (S x)) nopathstruetofalse).
Defined.

Lemma negpathssx0 (x : nat) : ¬ (S x = 0).
Proof.
  intros x X. apply (negpaths0sx x (pathsinv0 X)).
Defined.

Lemma invmaponpathsS (n m : nat) : S n = S m -> n = m.
Proof.
  intros n m e.
  set (f := λ n : nat, match n with O => O | S m => m end).
  apply (@maponpaths _ _ f (S n) (S m) e).
```

Library Monoids_and_Groups

Algebra I. Part B. Monoids, abelian monoids groups, abelian groups. Vladimir Voevodsky. Aug. 2011 - .

Contents

- Standard Algebraic Structures
  - Monoids
    - Basics definitions
    - Univalence for monoids
    - Functions between monoids compatible with structures (homomorphisms) and their properties
    - Subobjects
    - Quotient objects
    - Direct products
  - Abelian (commutative) monoids
    - Basic definitions
    - Univalence for abelian monoids
    - Subobjects
    - Quotient objects

Monoids

Basic definitions

```
Definition monoid : UU := total2 (λ X : setwithbinop, ismonoidop (@op X)).

Definition monoidpair :
  ∏ (t : setwithbinop), (λ X : setwithbinop, ismonoidop op) t → Σ X : setwithbinop, ismonoidop op :=
  tpair (λ X : setwithbinop, ismonoidop (@op X)).

Definition pr1monoid : monoid -> setwithbinop := @pr1 _ _.
Coercion pr1monoid : monoid >-> setwithbinop.

Definition assocax (X : monoid) : isassoc (@op X) := pr1 (pr2 X).

Definition unel (X : monoid) : X := pr1 (pr2 (pr2 X)).

Definition lunax (X : monoid) : islunit (@op X) (unel X) := pr1 (pr2 (pr2 (pr2 X))).

Definition runax (X : monoid) : isrunit (@op X) (unel X) := pr2 (pr2 (pr2 (pr2 X))).

Definition isasetmonoid (X : monoid) : isaset X := pr2 (pr1 (pr1 X)).

Notation "x + y" := (op x y) : addmonoid_scope.
Notation "0" := (unel _) : addmonoid_scope.

Delimit Scope addmonoid_scope with addmonoid.

Notation "x * y" := (op x y) : multmonoid_scope.
Notation "1" := (unel _) : multmonoid_scope.
```

Library PartB

Univalent Foundations. Vladimir Voevodsky. Feb. 2010 - Sep. 2011.

Port to coq trunk (8.4-8.5) in March 2014. The second part of the original uu0 file, created on Dec. 3, 2014.

This file starts with the definition of h-levels. No axioms are used. Only one universe UU is used, except in one case.

In the current approach to dependent eliminators for inductive types, in this case for nat , the type family appears as an argument of the eliminator in the form of a function. Therefore, if one wants to use the eliminator nat_rect to define a function nat -> UU one must use, as an argument, the function n : nat => UU whose type is forall n : nat, UU' where UU' is the type of UU . Therefore, the eliminator must be defined on arguments whose type contains UU' and we should acknowledge it as an instance of using a second universe.

Contents

- Basics about h-levels
  - h-levels of types
  - h-levels of functions
  - h-levelf of pr1

Basics about h-levels

h-levels of types

```
Fixpoint isofhlevel (n : nat) (X : UU) : UU
  := match n with
      | O => iscontr X
      | S m => ∏ x : X, ∏ x' : X, (isofhlevel m (x = x'))
      end.


Theorem hlevelretract (n : nat) {X Y : UU} (p : X -> Y) (s : Y -> X)
          (eps : ∏ y : Y , paths (p (s y)) y) : isofhlevel n X -> isofhlevel n Y.
Proof.
  intro.
  induction n as [ | n IHn ].
  - intros X Y p s eps X0. unfold isofhlevel.
    apply (iscontrretract p s eps X0).
  - unfold isofhlevel. intros X Y p s eps X0 x x'. unfold isofhlevel in X0.
    assert (is: isofhlevel n (paths (s x) (s x'))) by apply X0.
    set (s':= @maponpaths _ _ s x x'). set (p':= pathssec2 s p eps x x').
    set (eps':= @pathssec3 _ _ s p eps x x'). simpl.
    apply (IHn _ _ p' s' eps' is).
Defined.

Corollary isofhlevelweqf (n : nat) {X Y : UU} (f : X ≃ Y) :
  isofhlevel n X -> isofhlevel n Y.
Proof.
  intros n X Y f X0.
  apply (hlevelretract n f (invmap f) (homotweqinvweq f)).
  assumption.
```

Library PartC

Univalent Foundations. Vladimir Voevodsky. Feb. 2010 - Sep. 2011. Port to coq

trunk (8.4-8.5) in March 2014. The third part of the original uu0 file, created on Dec. 3, 2014.

Only one universe is used, and it is never used as a type.

The only axiom we use is funextemptyAxiom , which is the functional extensionality axiom for functions with values in the empty type. Any results that don't depend on axioms should be in an earlier file.

Contents

- More results on propositions
- Isolated points and types with decidable equality
  - Basic results on complements to a point
  - Basic results on types with an isolated point
  - Weak equivalences and isolated points
  - Recomplement on functions
  - Standard weak equivalences between compl T t1 and compl T t2 for isolated t1 and t2

Isolated points and types with decidable equality.

Basic results on complements to a point

```
Definition compl (X : UU) (x : X) := Σ x', x != x'.
Definition complpair (X : UU) (x : X) := tpair (λ x' : X, x != x').
Definition pr1compl (X : UU) (x : X) := @pr1 _ (λ x' : X, x != x').

Lemma isinclpr1compl (X : UU) (x : X) : isincl (pr1compl X x).
Proof.
  intros. apply (isinclpr1 _ (λ x' : X, isapropneg _)).
Defined.

Definition compl_ne (X:UU) (x:X) (neq_x : neqPred x) := Σ y, neq_x y.

Definition compl_ne_pair (X : UU) (x : X) (neq_x : neqPred x) (y : X)
           (ne :neq_x y) :
  compl_ne X x neq_x := (y,,ne).

Definition pr1compl_ne (X : UU) (x : X) (neq_x : neqPred x)
           (c : compl_ne X x neq_x) :
  X := pr1 c.

Definition make_negProp {P : UU} : negProp P.
Proof.
  intros. exists (¬ P). split.
      - apply isapropneg.          - apply isrefl_logeq.
Defined.
```

Library PartD

Univalent Foundations. Vladimir Voevodsky. Feb. 2010 - Sep. 2011.

Port to coq trunk (8.4-8.5) in March 2014. The third part of the original uu0 file, created on Dec. 3, 2014.

Only one universe is used and never as a type.

Contents

- Sections of "double fibration" (P : T -> UU) (PP : ∏ t : T, P t -> UU) and pairs of sections
  - General case
  - Functions on dependent sum (to a total2)
  - Functions to direct product
- Homotopy fibers of the map ∏ x : X, P x -> ∏ x : X, Q x
  - General case
  - The weak equivalence between sections spaces (dependent products) defined by a family of weak equivalences (P x) ≃ (Q x)
  - Composition of functions with a weak equaivalence on the right
- The map between section spaces (dependent products) defined by the map between the bases f : Y -> X
  - General case

- Sections of families over an empty type and over coproducts
  - General case
  - Functions from the empty type
  - Functions from a coproduct
- Sections of families over contractible types and over total2 (over dependent sums)
  - General case
  - Functions from unit and from contractible types
  - Functions from total2
  - Functiosn from direct product
- Theorem saying that if each member of a family is of h-level n then the space of sections of the family is of h-level n.
  - General case
  - Functions to a contractible type
  - Functions to a proposition
  - Functions to an empty type (generalization of isapropneg)
- Theorems saying that iscontr T , isweq f etc. are of h-level 1
- Theorems saying that various pr1 maps are inclusions
- Various weak equivalences between spaces of weak equivalences
  - Composition with a weak equivalence is a weak equivalence on weak equivalences
  - Invertion on weak equivalences as a weak equivalence
- h-levels of spaces of weak equivalences
  - Weak equivalences to and from types of h-level (S n)
  - Weak equivalences to and from contractible types
  - Weak equivalences to and from propositions
  - Weak equivalences to and from sets
  - Weak equivalences to an empty type

Sections of "double fibration" (P : T -> UU) (PP : Π t : T, P t -> UU) and pairs of sections

General case

```
Definition totaltoforall {X : UU} (P : X -> UU) (PP : ∏ x : X, P x -> UU) :
  total2 (fun s0 : ∏ x : X, P x
            => ∏ x : X, PP x (s0 x)) -> ∏ x : X, total2 (PP x).
Proof.
  intros X P PP X0 x.
  exists (pr1 X0 x).
  apply (pr2 X0 x).
Defined.

Definition foralltototal {X : UU} (P : X -> UU) (PP : ∏ x : X, P x -> UU) :
  (∏ x : X, total2 (PP x))
  -> total2 (fun s0 : ∏ x : X, P x => ∏ x : X, PP x (s0 x)).
Proof.
  intros X P PP X0.
  exists (λ x : X, pr1 (X0 x)).
  apply (λ x : X, pr2 (X0 x)).
Defined.

Theorem isweqforalltototal {X : UU} (P : X -> UU) (PP : ∏ x : X, P x -> UU) :
  isweq (foralltototal P PP).
Proof.
  intros.
  simple refine (isweq_iso (foralltototal P PP) (totaltoforall P PP) _ _).
  - apply idpath.
```

Homotopy fibers of the map Π x : X, P x -> Π x : X, Q x

General case

```
Definition maponsec {X : UU} (P Q : X -> UU) (f : ∏ x : X, P x -> Q x) :
  (∏ x : X, P x) -> (∏ x : X, Q x)
  := fun s : ∏ x : X, P x => (λ x : X, (f x) (s x)).

Definition maponsec1 {X Y : UU} (P : Y -> UU) (f : X -> Y) :
  (∏ y : Y, P y) -> (∏ x : X, P (f x))
  := fun sy : ∏ y : Y, P y => (λ x : X, sy (f x)).

Definition hfibertoforall {X : UU} (P Q : X -> UU) (f : ∏ x : X, P x -> Q x)
           (s : ∏ x : X, Q x) :
  hfiber (@maponsec _ _ _ f) s -> ∏ x : X, hfiber (f x) (s x).
Proof.
  intro. intro. intro. intro. intro. unfold hfiber.
  set (map1 := totalfun (fun pointover : ∏ x : X, P x =>
                           paths (λ x : X, f x (pointover x)) s)
                        (fun pointover : ∏ x : X, P x =>
                           ∏ x : X, paths ((f x) (pointover x)) (s x))
                        (fun pointover: ∏ x : X, P x =>
                           toforallpaths _ (λ x : X, f x (pointover x)) s)).
  set (map2 := totaltoforall P (λ x : X,
                                  (λ pointover : P x,
                                     paths (f x pointover) (s x)))).
  set (themap := λ a : _, map2 (map1 a)). assumption.
Defined.
```

Library UnivalenceAxiom

The univalence axiom and its consequences

In this file, we formulate univalence and its consequences, including functional extensionality (the statement that functions with equal values are equal).

One approach would be to take univalence as the only axiom and to formulate theorems proving the consequences, whose proofs would appeal to the univalence axiom. We adopt a different approach here, preferring also to introduce axioms for various consequences of univalence. This allows us to measure how subsequent theorems depend on the axioms using the "Print Assumptions" command of Coq.

An important point is that the types of all our axioms are propositions, that is types of h-level 1 and once an element of such a type is declared or contructed it becomes a contractible type.

In particular, the declared element corresponding to the axiom and the element that can be contructed using another, stronger, axiom are connected by a path.

Proofs that the types of our axioms are propositions will be provided in the file UnivalenceAxiom2.

We postpone stating the axioms themselves until after all the implications are established; this helps us encourage the use of the axioms for the consequences of univalence, rather than the theorems giving the implications.

We show that univalenceAxiom is equivalent to the axioms weqtopathsStatement and weqpathsweqStatement stated below .

```
Theorem univfromtwoaxioms :
  (Σ weqtopaths: weqtopathsStatement, weqpathsweqStatement weqtopaths)
  <-> univalenceStatement.
Proof.
  split.
  { intros [weqtopaths weqpathsweq] T1 T2.
    set ( P1 := λ XY : UU × UU, pr1 XY = pr2 XY ) .
    set ( P2 := λ XY : UU × UU, (pr1 XY) ≃ (pr2 XY) ) .
    set ( Z1 := total2 P1 ). set ( Z2 := total2 P2 ).
    set ( f := totalfun _ _ ( λ XY : UU × UU, @eqweqmap (pr1 XY) (pr2 XY)) : Z1 -> Z2 ) .
    set ( g := totalfun _ _ ( λ XY : UU × UU, weqtopaths (pr1 XY) (pr2 XY) ) : Z2 -> Z1 ) .
    assert (efg : funcomp g f ~ idfun _) .
    - intro z2 . induction z2 as [ XY e ] .
      unfold funcomp . unfold idfun . unfold g . unfold f . unfold totalfun . simpl .
      apply ( maponpaths ( fun w : ( pr1 XY) ≃ (pr2 XY) => tpair P2 XY w )
                         ( weqpathsweq ( pr1 XY ) ( pr2 XY ) e )) .
    - set ( h := λ a1 : Z1, pr1 ( pr1 a1 ) ) .
      assert ( egf0 : ∏ a1 : Z1 , pr1 ( g ( f a1 ) ) = pr1 a1 ).
      + intro. apply idpath.
      + assert ( egf1 : ∏ a1 a1' : Z1 , paths ( pr1 a1' ) ( pr1 a1 ) -> a1' = a1 ).
        * intros.
          set ( X' := maponpaths pr1 X ).
          assert ( is : isweq h ).
          { simpl in h . apply isweqpr1pr1 . }
          apply ( invmaponpathsweq ( weqpair h is ) _ _ X' ).
        * set ( egf := λ a1 , egf1 _ _ ( egf0 a1 ) ).
          set ( is2 := isweq_iso _ _ egf efg ).
          apply ( isweqtotaltofib _ _ ( λ _, eqweqmap) is2 ( dirprodpair T1 T2 ) ).
          }
```

数学とNP問題

「数学とNP問題」というのは、少し奇妙なタイトルかもしれない。

というのも、PやNPといったクラスを扱う「複雑性理論」自身が、数学の一分野だからである。正確にいうと、複雑性理論は、数学理論であると同時に、現在では、物理理論の一部でもある（もちろん、計算機科学の一分野でもある）。

ここでは、数学理論としての複雑性の理論に、問題を限ってみよう。

次に述べることは、冒頭のタイトルに関係するが、少し、別のことである。

数学の問題Xがあったとしよう。問題Xを「解く」ということは、基本的には、ある形式的体系Tを選んで、その中で、Xの「証明」Sを、「数学的に」与えることである。

でも、問題Xの証明S（だと、当人が主張しているもの）が、正しい保証はない。世の中には、「間違った証明」が山ほどある。天才Wilesにしろ天才Voevodskyにしろ、自分が「間違った証明」を論文として公開したことがあることを認めているくらいだ。

それでは、ある問題Xのある証明Sが、「正しい」ことは、どのように検証できるのか？　あるいは、もっと原理的に考えて、それが「検証可能」であるという主張には、どのような前提が潜んでいるのか？

一つの考え方は、数学の証明は、「多項式時間」で「検証可能」でなければならないと考えることだ。（今の僕は、それが正しいと感じているのだが。）

ただ、こうした考えに立って、数学の証明の「正しさ」が、多項式時間で「検証可能」だということは、数学の証明可能な問題全体が、NPクラスに属するということにならないか？

な、わけはない。（とも、思うのだが。）

僕が感じている「違和感」の正体を、もう少し詳しく考えたいと思う。

僕の興味は、人間と機械の認識能力の階層性（感覚・運動的外界の把握の階層、言語能力による世界把握の階層、数学的認識能力による宇宙の把握の階層）と、それぞれの階層の認識能力の「限界」にある。

言語能力が、クラスPに属するだろうということは、前に書いた。数学的能力が、全体としてクラスNPもどきの構造をしているように見えるのは、面白いことだ。それは、二つの能力が、別の階層に属することをしめしている。

複雑性理論でのクラス
MA: Merlin-Arthur

Merlin-Arthur クラス

- 数学の証明について、次のような複雑性のクラスを考える
- Mは超優秀だが、あまり信用のおけないところもある数学者。多項式時間でチェックできる「証明」を提出する。
- Aも、やはり数学者。Mが提出した「証明」を多項式時間でチェックする。
- クラス MA は、ある言語Lのクラスで、全てのxについて、次のようなアルゴリズムが存在するというクラスとして定義する。

1. If $x \in L$, then there exists at least one certificate w such that $V(x,w)$ accepts with certainty.
2. If $x \notin L$, then regardless of w, $V(x,w)$ rejects with probability at least ½.

- NPもBPPもクラスMAに含まれる。また、MAはPPに含まれる。

Arthur-Merlin クラス

- 今度は、次のような状況を考える。
- 実直な数学者Aは、いい加減な（でも超優秀な）数学者Mに、ある問題を解くように依頼する。Mはこの依頼に応えなければならない。こうした問題のクラスを、AMという。
- このクラスAM, MAの関係は、よくわかっていない。でも、多くの人は、AM = MA = NP だと予想している。

まとめにかえて

「機械は考えることができるか？」

- 「人工知能論」の中心的問題は、「機械は考えることができるか？」という問題である。人工知能研究の発展は、この問題に肯定的に、正面から応えることによって可能となる。
- 確かに、70年前にチューリングが予想したように、この問題についての「人々の意識」は大きく変わった。ただ、そのことは、人工知能研究が本質的な前進を果たしたことを、必ずしも意味しはしない。
- 生物学的進化と人類史的進化・文明史的進歩の中で形成された人間の知能の階層構造の中で、重要な「言語能力」「数学的能力」に対する取り組みは、これからの課題である。

人工知能における計算主義的アプローチ

- 人工知能における計算主義的アプローチは、人間の知能の中核を「計算する能力」として捉える。人間と機械を、「計算」という抽象のレベルで「同一視」する計算主義は、強力な方法論である。
- 人間の「数学的能力」においては、その中核が「計算する能力」であることは、ある意味自明なのだが、それ以外の多様な人間の諸能力においても、例えば、「言語能力」や記憶の外延的拡大である「知識グラフ検索」においても、計算主義的アプローチは有効である。

人間と機械の認識の「限界」

- 「人工知能に何ができ、何ができないのか？」という問いを、「人間には何ができ、機械には何ができないのか？」という形の問いにしてはいけない。
- 人工知能にできることには限界がある。その限界について考えることも、人工知能研究の重要な課題だ。ただ、その限界について知ることは、計算主義的な人工知能観に立つ限りは、人間と機械の双方がともに持つ認識の限界について知ることである。それは、「計算」の限界について知ることだ。
- 人工知能は、その限界に従う。計算の限界については、計算の複雑性の理論が、多くのことを教えてくれる。

計算可能性理論と計算複雑性理論

- 全く自由に見える数学においても、そこで「証明可能＝計算可能」なものには限界がある。このことは、1950年代には、「計算可能性理論」として、一般的に認められるようになった。ただ、それは、我々の「限界」としては、非常に荒い、かつ、抽象的で原理的な限界を与えるものでしかなかった。
- 基本的には、多項式時間で計算できるものを「易しい計算」、指数関数的時間でしか計算できないものを「難しい計算」と考える「計算複雑性理論」が普及するのは、1970年代にはいってからだ。それは、計算の限界について、より具体的な判断基準を与えるものだ。
- ただ、計算複雑性理論の基本問題である、「P ＝ NP ？ 問題」（それは、「難しい計算」が「易しい計算」には還元できないことを主張する）が未解決のままなこともあり、この理論と人工知能論との接点は、大きなものには広がらなかった。

計算可能性概念の「物理化」

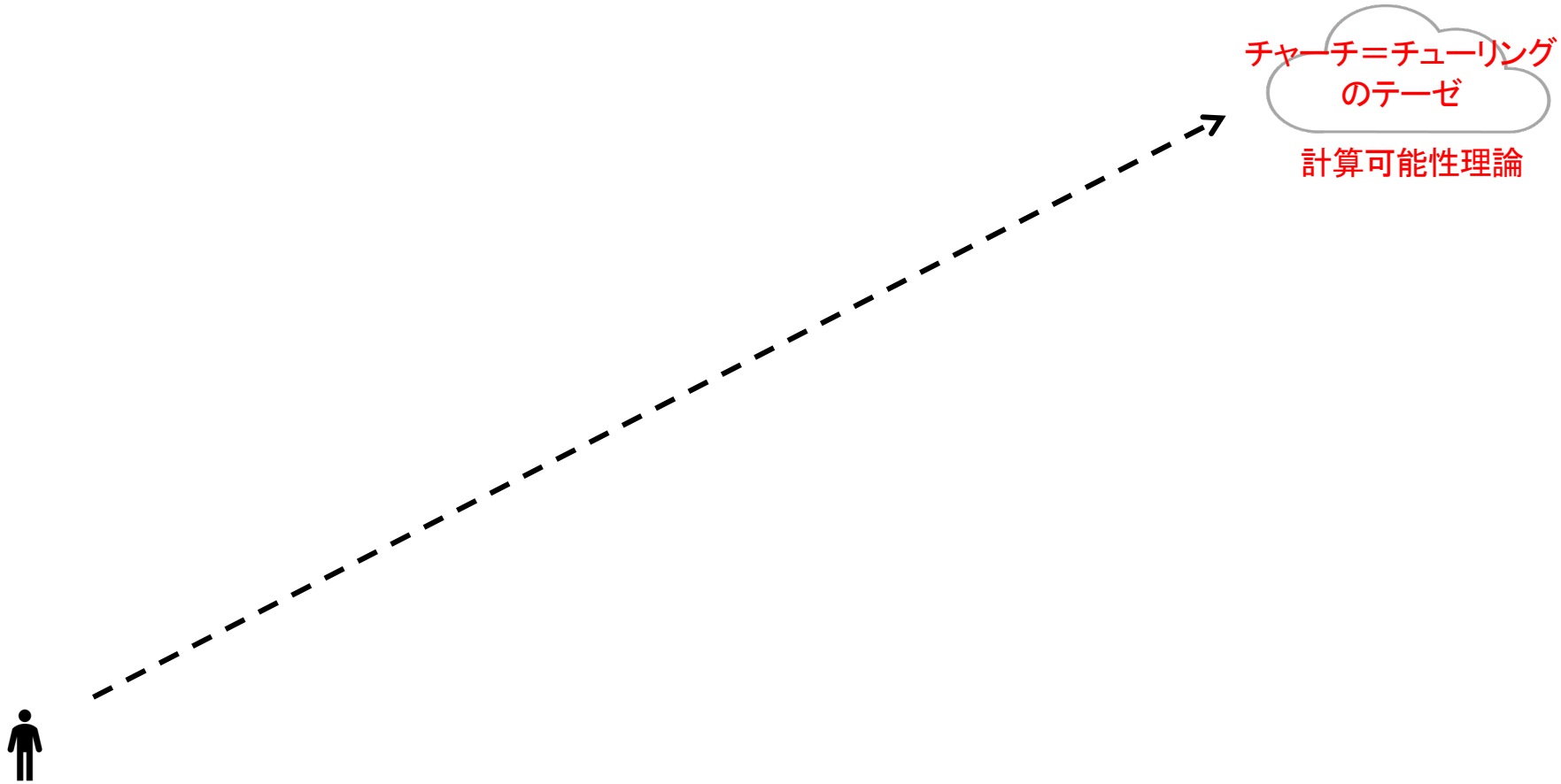
- 80年代に入っての大きな変化は、計算可能性概念が、「物理化」されたことだ。実現されるべき人工知能は、物理的なものでなければいけないのだから、計算主義的人工知能論にとって、この変化は重要である。
- しかし、この「物理化」は、概念のレベルで抽象的に行われたものである。量子コンピュータは、まだ、物理学者の頭の中にしか存在しなかった。
- 面白いのは、計算可能性概念より、計算の限界の精緻な境界を与える計算複雑性の理論は、すぐにはこの「物理化」の影響を受けなかったことだ。
- これらの三つの理論は、現実の人間からも、人工知能論の理論的枠組み（コネクショニズムの全盛期だ）からも、遠く離れたところにあった。

複雑性理論の発展

- 90年代に入って、量子複雑性の理論化が始まる。量子チューリング・マシンで多項式時間で計算可能な複雑性のクラスBQPに、素因数分解のアルゴリズムが含まれることは、大きな発見だった。
- 21世紀に入ると、物理学の世界で、量子情報理論と複雑性理論への関心が急速に拡大する。

- 計算の「限界」は、我々の身近の具体的な問題に近づいてきている。物理学の複雑性理論の発展もめざましい。

50年代



「証明可能＝計算可能」なものには限界があることは、
1950年代には、一般に認められるようになった。
ただ、それは、我々の認識の「限界」としては、非常に
荒い、かつ、抽象的で原理的な限界を与えるものでしかなかった。

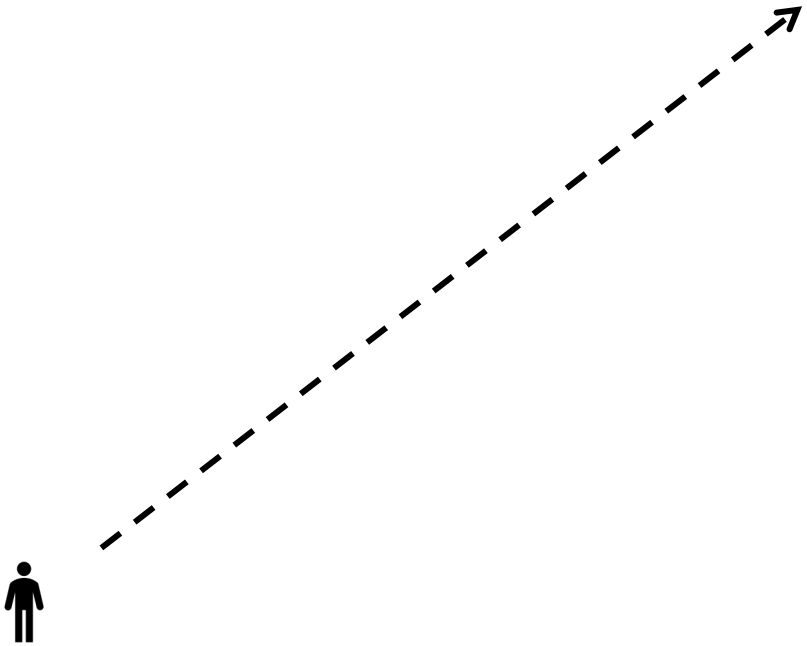
70年代

チャーチ＝チューリング
のテーゼ

計算可能性理論

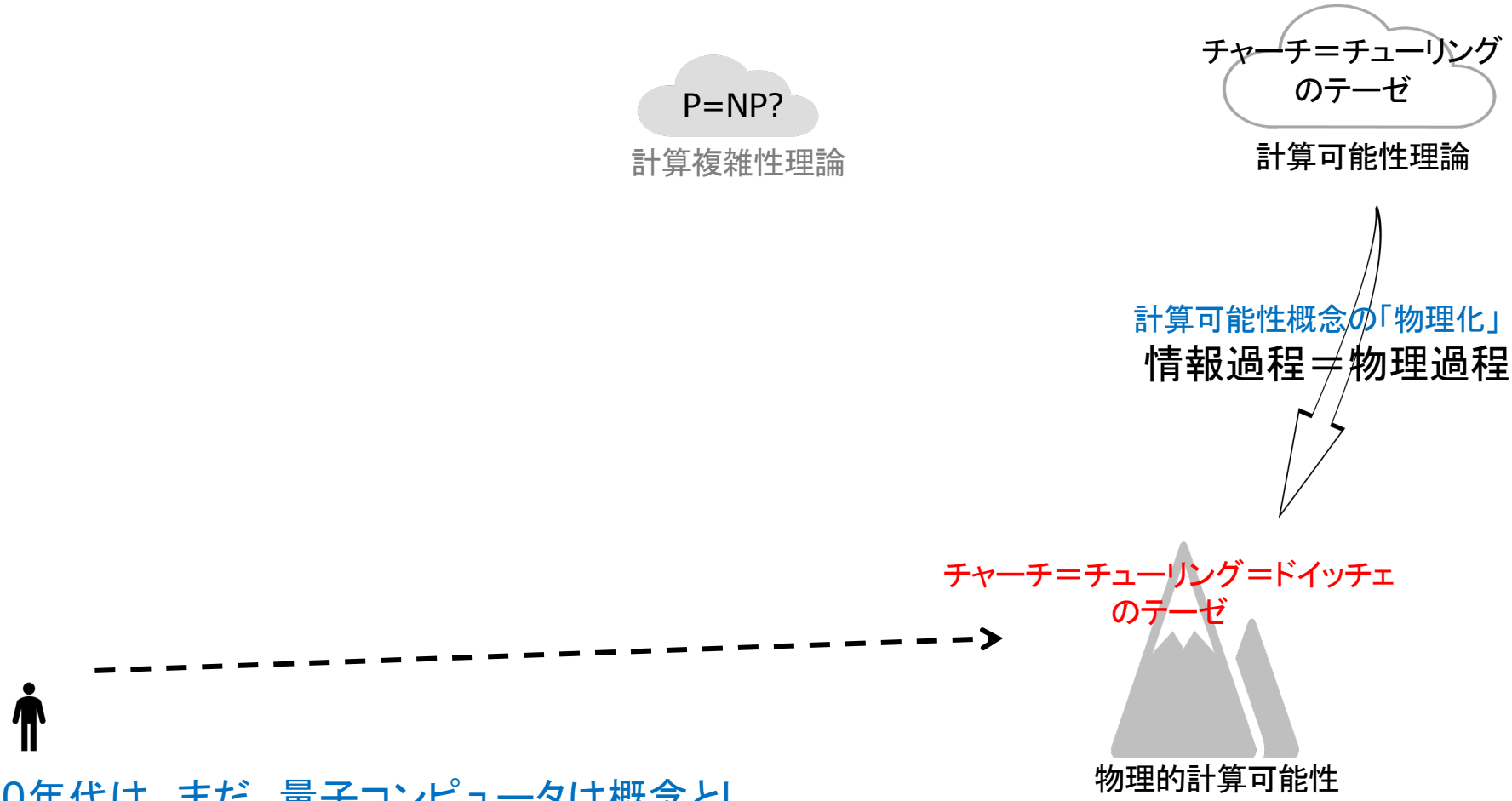
P=NP?

計算複雑性理論



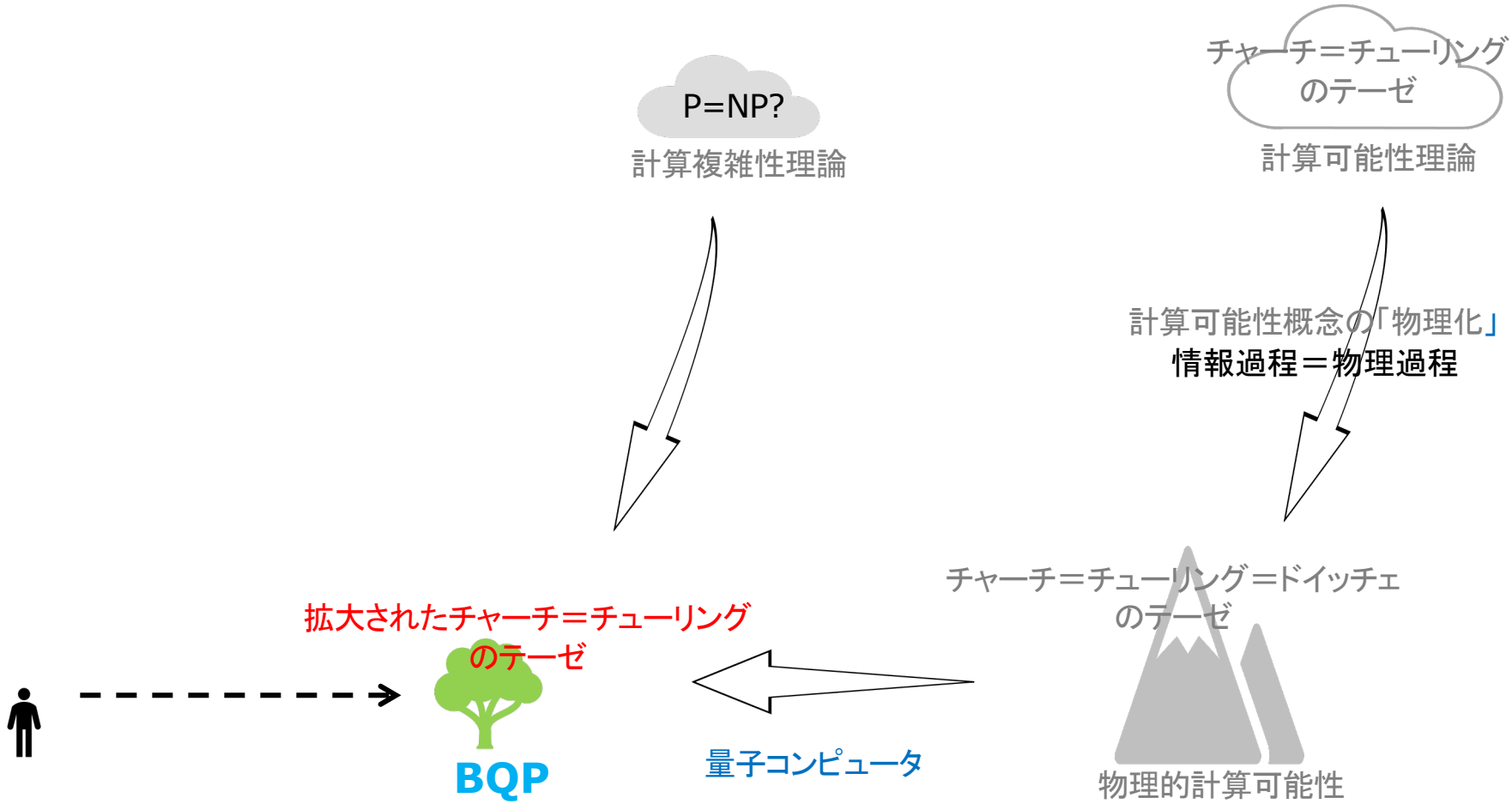
ただ、計算複雑性理論の基本問題である、「P ＝ NP ？ 問題」（それは、「難しい計算」が「易しい計算」には還元できないことを主張する）が未解決のままなこともあり、この理論と人工知能論との接点は、大きなものには広がらなかった。

80年代



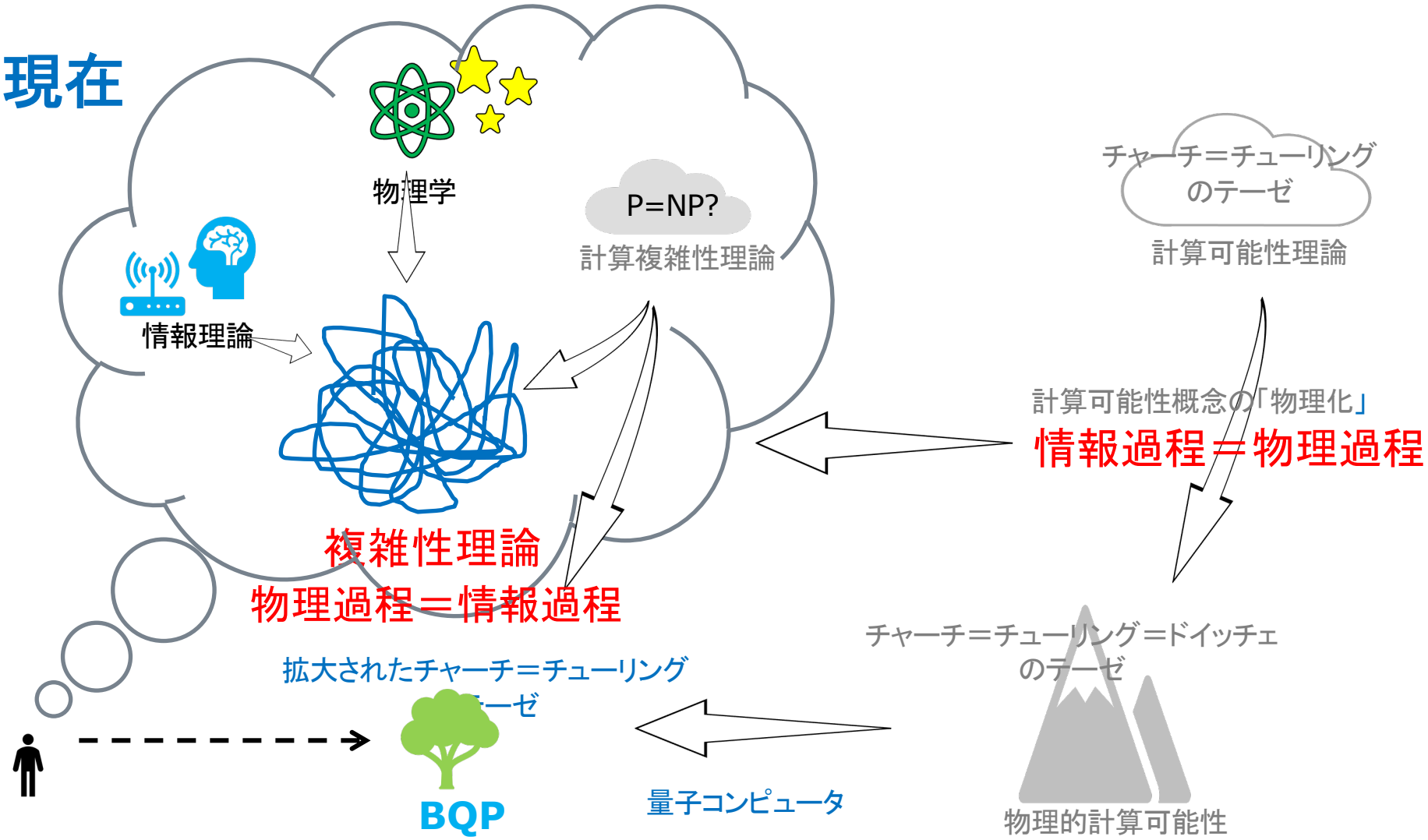
80年代は、まだ、量子コンピュータは概念としてしか存在していなかった。「計算可能性」の物理化という画期的な変化も、まだまだ、抽象的な議論だった。

90年代



量子複雑性の理論が登場し、BQPという新しい複雑性の概念が確立する。ショアのアルゴリズムは、広い関心をあつめた。

現在



計算の「限界」は、我々の身近の具体的な問題に近づいてきている。物理学の複雑性理論の発展もめざましい。

我々の認識の「限界」の認識が進む一方で、我々の世界の認識が拡大する

- 複雑性理論は、かつての数学的・抽象的な認識の限界の理論（「チャーチ＝チューリングのテーゼ」）から、物理的・具体的ではあるが、観念の中にしかないものを対象にした限界の理論（「チャーチ＝チューリング＝ドイッチェのテーゼ」）へと発展してきた。かつ、それは、量子コンピュータが現実味を増すにつれて、より具体的な量子複雑性のもとでの限界の理論（「拡大されたチャーチ＝チューリングのテーゼ」）へと変化してきた。
- それは、我々の認識の「限界」の認識が、より一層、精緻で現実的なものに進んだということである。
「我々が知りうることは、限られている。」
- その一方で、そうした限界の認識の深化は、我々の世界の認識が拡大する過程と一体なのだ！
「我々が知りうることは、拡大している。」

機械と人間の関係を、改めて考える

- 量子複雑性の問題を考えると、機械と人間の関係で新しい問題が提起されていることに気づく。今までは、人間の知能の本質を「計算」として、その抽象の下では、機械と人間の等値が可能だと論じてきた。ただ、BQP問題を解く能力は、機械に属し、人間には属さない！
- ペンローズのように、人間の知能の根底には、量子論的な過程があるとする人工知能論は既に存在する。ただ、ペンローズが無邪気に想定していたように、量子過程は、人間の知能の謎を全て解くほど万能ではないのだ。NP完全問題は、量子コンピュータでも解決不能である。そのことを、量子複雑性の議論は教えてくれる。
- ただ、機械と人間の関係を、こうした知見の下で、改めて考える必要はある。

人間と機械の系として、人工知能を考える

- 人工知能技術を「機械に何ができるか？」という視点でだけ捉えるのは、狭いのだと思う。この分野で重要なことは、「機械に何ができるか？」ということではなく、「機械と人間で何ができるのか？」ということではないかと考えている。複雑性理論での、MAあるいはAM問題の捉え方は示唆的である。どちらかを人間にどちらかを機械と考えればいい。
- 実際、COQのような「証明支援システム」をつかってすぐに気づくのは、それは人間の証明を機械が支援するシステムなのだが、それは、機械の証明を人間が支援するシステムでもあるということだ。ここでは、人間と機械が「共生」している。
- 既に、物理・科学の世界では、機械はなくてはならない、我々の認識の「同伴者」である。そこで得られた認識は、人間のものである。ただ、機械の助けなしには、我々は、そうした認識を獲得できない。

言語的認識能力の解明が重要な課題

- 数学的認識の解明は、なかなか難しい課題だと感じている。

- 人工知能研究で、言語能力の解明に力が注がれるのは間違い無い。言語が人間の認識活動の中で示す大きな役割を見れば、それは、当然である。（ただ、それには言語研究者とAI研究者の協力が不可欠だと僕は考えている。）
- 言語能力が万人に与えられてはいるが、個人の能力として現れるのに対して、数学的・科学的認識は、我々人類全体に帰属すべきものではあるが、個人の能力を超えている。

  こうしたギャップを、人工知能技術が補完すると考えるのは楽しいことだ。